

**SISTEM TEMU KEMBALI INFORMASI DENGAN
MENERAPKAN METODE *PROBABILISTIK BINARY
INDEPENDENCE MODEL* (BIM).**

TUGAS AKHIR

oleh :

FITRO AIDIL PURNAMA
10751000128



**JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU
2012**

SISTEM TEMU KEMBALI INFORMASI DENGAN MENERAPKAN METODE PROBABILISTIK BINARY INDEPENDENCE MODEL (BIM)

FITRO AIDIL PURNAMA
10751000128

Tanggal Sidang : 22 November 2012
Periode Wisuda : 28 Februari 2013
Jurusan Teknik Informatika
Fakultas Sains Dan Teknologi
Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Information retrieval merupakan salah satu solusi bagi para pencari informasi untuk mendapatkan informasi yang dibutuhkan. Hal yang menjadi kepentingan dalam *information retrieval* adalah nilai relevansi antara *query* dan *corpus* yang tersedia. Pada penelitian ini, Sistem Information Retrieval dibangun dengan menggunakan Metode *Binary Independence Model* (BIM). Metode BIM ini berfungsi untuk mengetahui nilai *relevansi* suatu dokumen yang dicari berdasarkan pembobotan *biner* yang disesuaikan dengan *query* yang diinputkan. Pada sistem ini, *corpus* yang disediakan adalah sebanyak 281 dokumen yang merupakan dokumen berita dalam bahasa Indonesia. Seluruh dokumen yang terdapat dalam *korpus* ini akan diindeks terlebih dahulu dan disimpan dalam file *token.txt*, kemudian pengguna yang menginputkan *query* akan memperoleh hasil pencarian yang telah teranking oleh sistem sesuai dengan nilai *relevansi* antara *query* dan dokumen ter-*retrieve*. Dari hasil pengujian yang dilakukan, diperoleh kesimpulan bahwa korelevanan dokumen yang dicari pada dasarnya tergantung pada pengguna, namun metode (*BIM*) yang telah diteliti ini dapat mencari dokumen yang *relevan* dan tidak *relevan*, nilai *relevansi* yang diperoleh setelah menggunakan *stemming* mengalami peningkatan, yaitu sebesar 50% berdasarkan pengujian *precision and recall*.

Kata kunci : *Binary Independence ,corpus, information retrieval,Model, query Relevan*

DAFTAR ISI

	Halaman
ABSTRAK	i
ABSTRACT	ii
DAFTAR ISI	iii
DAFTAR GAMBAR	iv
BAB I PENDAHULUAN	
1.1. Latar Belakang	I-1
1.2. Rumusan Masalah	I-2
1.3. Batasan Masalah.....	I-3
1.4. Tujuan Penelitian.....	I-3
1.5. Sistematika Penulisan	I-3
BAB II LANDASAN TEORI	
2.1. <i>Sistem Temu Kembali Informasi</i>	II-1
2.2. Arsitektur Sistem Temu Kembali Informasi	II-2
2.3. Pembuatan Sistem Temu Kembali Informasi.....	II-3
2.3.1. <i>Stemming</i>	II-4
2.3.1.1 Algoritma Porter	II-4
2.3.1.1.1 Porter <i>Stemmer</i> Bahasa Indonesia	II-5
2.3.1.2 Algoritma Nazief & Adriani	II-6
2.4. Model Probabilistik.	II-9
2.4.1. <i>Binary Independence model (BIM)</i>	II-10
2.5. Model Boolean	II-10
2.6. Model Ruang Vektor (<i>Vektor Space Model-VSM</i>)	II-11
2.7. <i>Recall</i> dan <i>Precision</i>	II-13

BAB III METODOLOGI PENELITIAN

3.1. Identifikasi Masalah.....	III-2
3.2. Tahap Pengumpulan Data	III-2
3.3. Analisa	III-2
3.3.1 Analisa BIM untuk Sistem temu kembali informasi.....	III-2
3.3.2 Analisa Sistem	III-3
3.4. Perancangan Sistem.....	III-3
3.5. Implementasi Sistem.....	III-3
3.6. Pengujian Sistem	III-4
3.7. Kesimpulan dan Saran	III-4

BAB IV ANALISA DAN PERANCANGAN

4.1. Analisa Sistem Temu Kembali Informasi dengan <i>Binary Independence Model</i>	IV-1
4.1.1 Koleksi Dokumen	IV-2
4.1.2 <i>Text Operation</i> terhadap koleksi dokumen.....	IV-2
4.1.3 <i>Indexing</i>	IV-6
4.1.4 Pembobotan Kata (<i>Term</i>) Hasil <i>Indexing</i>	IV-7
4.1.5 Pembobotan Kata Pada <i>Query</i>	IV-8
4.1.6 Perhitungan nilai relevansi dokumen terhadap <i>Query</i> dan perangkingan dokumen	IV-9
4.2 Analisa sistem.....	IV-10
4.2.1 <i>Flowchart</i>	IV-10
4.2.2. <i>Use case diagram</i>	IV-12
4.3. Perancangan.....	IV-13
4.3.1. Perancangan File teks (<i>Flat file</i>).....	IV-14
4.3.2. Perancangan Antarmuka (<i>Interface</i>) Sistem.....	IV-15

BAB V IMPLEMENTASI DAN PENGUJIAN

5.1. Implementasi	V-1
5.1.1. Batasan implementasi.....	V-1

5.1.2. Lingkungan Implementasi	V-1
5.1.3. Hasil Implementasi	V-2
5.2. Pengujian Sistem	V-8
5.2.1. Hasil Pengujian Menggunakan <i>Stemming</i>	V-9
5.2.2. Hasil Pengujian Tanpa Menggunakan <i>Stemming</i>	V-10
5.2.3. Pengujian Tipe Dokumen	V-12
5.2.4. Kesimpulan Pengujian.....	V-12
BAB VI PENUTUP	
6.1. Kesimpulan.....	VI-1
6.2. Saran	VI-1
DAFTAR PUSTAKAvi
LAMPIRAN A	A-1
LAMPIRAN B.....	B-1
LAMPIRAN C.....	C-1
LAMPIRAN D	D-1

BAB I

PENDAHULUAN

1.1 Latar belakang

Information Retrieval merupakan bagian dari *computer science* yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dokumen itu sendiri dengan melakukan pencarian informasi dokumen berdasarkan teknik-teknik tertentu. Setiap hari orang mencari informasi dengan mengetikkan kata kunci pada sebuah mesin pencarian dan menginginkan informasi yang cepat dan akurat. Sistem temu kembali informasi ini pada dasarnya diambil dari pencarian pada database yang membagi data dalam sebuah *field-field* tertentu.

Pada dasarnya perkembangan sistem temu kembali informasi ini sebenarnya tidak dapat dipisahkan dari teknik atau metode-metode yang dipakai untuk menemukan dokumen-dokumen yang dicari. Salah satu metode yang dipakai adalah metode *Binary Independence Model (BIM)*. Metode *Binary Independence model* merupakan salah satu metode *Information Retrieval* yang menerapkan salah satu model klasik IR, yaitu model *probabilistic*. Model *probabilistic* memodelkan setiap kata dalam sebuah dokumen sebagai jawaban dari setiap kata dalam *query* yang ingin dicari. Setiap kata dalam *query* dianggap pasti akan memiliki kata yang sama dalam dokumen. Dengan kalimat lain, setiap kata dalam *query* diinisialisasi memiliki kemungkinan pasti ada dengan dokumen yang tersedia dan setiap kemungkinan tersebut akan diindeks yang disebut dengan *term frequency (tf)* sesuai banyaknya kata pada sebuah dokumen yang dikenal dengan istilah *inverted document frequency (idf)* sebagai acuan perankingan dokumen yang akan ditampilkan. Hal ini menjelaskan bahwa kemampuan lain dari model ini selain mampu melakukan perankingan

dokumen, model ini juga mampu melakukan *partial matching query* dengan dokumen yang dianggap sesuai. (Fuhr,1992)

Berbeda dengan dua model IR lainnya, yaitu model Boolean yang memodelkan dokumen layaknya sebuah himpunan dan query sebagai pernyataan Booleannya, kemudian diproses dengan operator-operator Boolean. Layaknya output dari operator Boolean yang hanya akan menghasilkan dua kemungkinan output, yaitu *true* dan *false*. Hal ini terkesan lebih sederhana dan lebih cepat dibandingkan dengan model *probabilistic*, namun output (dokumen) yang akan diperoleh hanya dokumen yang benar-benar sama dengan *query* yang diberikan (*true*) sehingga sistem tidak dapat memberikan peringkat (*rank*) dokumen yang terambil dan tidak mampu melakukan *partial matching* pada dokumen yang tersedia (Manning, 2009). Model lainnya adalah model vektor. Model ini memiliki pendekatan pemodelan dokumen seperti pada model *probabilistic*. Namun, model ini hanya mampu melakukan pengindeksan dokumen yang berdekatan saja untuk mengindeks dokumen, artinya model ini membutuhkan perhitungan jarak antara vektor dokumen dan *keyterm*-nya sehingga model ini dapat memberikan kemungkinan terjadinya tidak adanya dokumen yang bisa ditampilkan dalam satu pencarian (Manning,2009)..

Dari penjelasan di atas, maka diangkatlah metode BIM yang pada dasarnya menggunakan konsep dari model *probabilistic* pada penelitian ini yang memang memiliki kelebihan dari model-model lainnya. Metode BIM bekerja dengan cara melakukan pengecekan *corpus* (koleksi dokumen) terhadap query yang diberikan melalui pengindexan berdasarkan *term frequency* pada *query*. Kemudian dilakukan penghitungan *inverted document frequency* berdasarkan *term frequency* untuk proses pembobotan, hingga diperoleh rank dokumen yang paling relevan berdasarkan query yang diberikan (Cios,2007).

Satu hal yang menarik bagi para peneliti terhadap metode BIM ini adalah hasil penelitian Thomas Roelleke dan Jun Wang (2007) yang meneliti metode BIM menggunakan database SQL sebagai tempat penyimpanan data lalu melakukan pencarian dokumen yang *relevan* sesuai kebutuhan dengan penerapan Metode *BIM*.

Selain itu, korpus yang digunakan dalam penelitian-penelitian yang ada rata-rata menggunakan *stemming* dalam bahasa inggris. Dan masih banyak lagi jurnal-jurnal international lainnya yang meneliti tentang information retrieval.

Oleh sebab itu, penelitian ini mencoba membahas tentang penerapan metode *Binary Independence Model* dalam Sistem Temu kembali informasi.

1.2 Rumusan Masalah

Dari latar belakang diatas, dapat dirumuskan permasalahan yaitu Bagaimana merancang sistem pencarian dokumen meggunakan metode *Binary Independece Model* dengan mengikuti tahapan-tahapan yang terdapat dalam Sistem Temu Kembali Informasi dengan menggunakan *corpus* berbahasa indonesia.

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah :

1. Koleksi dokumen (*corpus*) yang digunakan hanya yang berekstensi (*extension*) .htm, .html dan .txt yang biasa digunakan untuk penelitian di bidang IR.
2. Dokumen yang digunakan adalah dokumen yang berbahasa Indonesia.
3. Dokumen yang digunakan ada 281 dokumen.
4. *Stemming* yang digunakan adalah algoritma *stemming* nazief dan adriani

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah dapat menerapkan metode *Binary Independence Retrieval* pada Sistem Temu Kembali Informasi dalam menemukan dokumen yang relevan dan tidak relevan dalam *corpus* Bahasa Indonesia.

1.5 Sistematika Penulisan

Laporan tugas akhir ini terdiri dari enam bab, dengan sistematika penulisan tersusun atas bab:

I. Pendahuluan

Membahas mengenai latar belakang permasalahan, rumusan masalah, batasan masalah, tujuan pembahasan, dan sistematika penulisan.

II. Landasan Teori

Membahas mengenai teori-teori yang berhubungan dengan pembahasan tugas akhir ini. Teori yang diangkat yaitu mengenai Sistem Temu Kembali Informasi.

III. Metodologi Penelitian

Membahas tahapan penelitian yaitu, identifikasi masalah, perumusan masalah, *study literatur*, analisa sistem, perancangan sistem, implementasi sistem, pengujian sistem, dan kesimpulan akhir.

IV. Analisa dan Perancangan

Membahas tentang analisa alur operasi *information retrieval* dengan metode *Binary Independence Model*, transformasi koleksi dokumen ke *korpus* merancang antarmuka sistem (*interface*), dan merancang sistem temu kembali informasi dengan metode *Binary Independence Model*.

V. Implementasi dan Pengujian

Membahas mengenai implementasi dan pengujian terhadap sistem temu kembali informasi dengan menggunakan metode *Binary Independence Model*, serta kesimpulan dari pengujian.

VI. Penutup

Bab ini berisikan kesimpulan dari tugas akhir yang dibuat dan saran-saran penulis kepada pembaca.

BAB II

LANDASAN TEORI

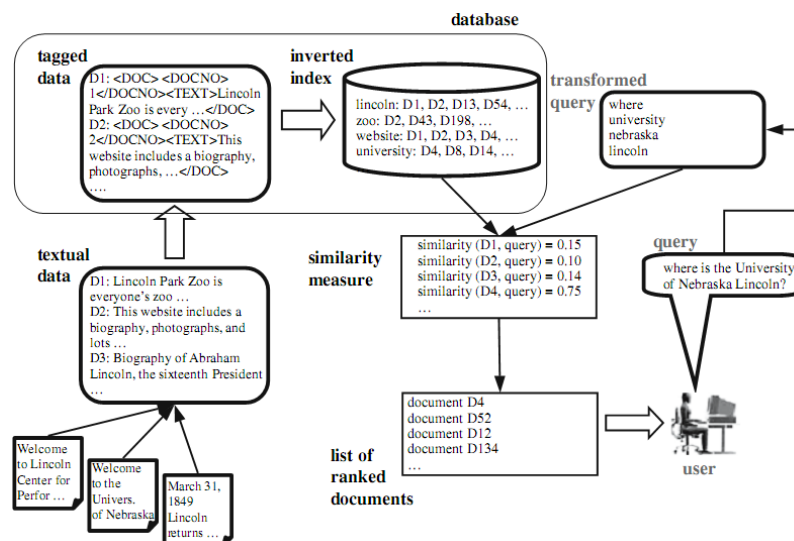
Penyusunan tugas akhir ini membahas mengenai Sistem Temu Kembali Informasi (*information retrieval*) dengan model ruang vektor. Sehingga pembahasan teori yang mendukung pelaksanaan tugas akhir ini dengan membahas teori *information retrieval* khususnya *Binary Independence Model* (BIM).

2.1 Sistem Temu Kembali Informasi (*Information Retrieval*)

Information Retrieval (IR) didefinisikan sebagai tindakan, metode dan prosedur untuk menemukan kembali data yang tersimpan, kemudian menyediakan informasi mengenai subyek yang dibutuhkan. Tindakan tersebut mencakup *text indexing*, *inquiry analysis*, dan *relevance analysis*. Data mencakup teks, tabel, gambar, ucapan, video, dan lainnya serta informasi termasuk pengetahuan terkait yang dibutuhkan untuk mendukung penyelesaian masalah dan akuisisi pengetahuan (Cios dkk, 2007). Tujuan utama dalam *information retrieval* adalah memenuhi kebutuhan informasi pengguna (*user*) dengan *me-retrieve* semua dokumen yang relevan, dan tidak *me-retrieve* dokumen yang tidak relevan. Dalam sistem IR digunakan fungsi heuristik untuk mendapatkan dokumen-dokumen yang relevan dengan *query* pengguna sehingga dokumen-dokumen tersebut dapat ditampilkan terurut berdasarkan nilai relevansinya terhadap *query* (perangkingan dokumen).

2.2 Arsitektur Sistem Temu Kembali Informasi

Secara garis besar arsitektur sistem Sistem Temu Kembali Informasi diperlihatkan pada Gambar 2.1. Ada dua pekerjaan yang ditangani oleh sistem ini, yaitu melakukan *pre-processing* terhadap database dan kemudian menerapkan metode tertentu untuk menghitung kedekatan (*relevansi* atau *similarity*) antara dokumen di dalam database yang telah di-*preprocess* dengan *query* pengguna. Pada tahapan *preprocessing*, sistem yang berurusan dengan dokumen *semi-structured* biasanya memberikan *tag* tertentu pada *term-term* atau bagian dari dokumen; sedangkan pada dokumen tidak terstruktur proses ini dilewati dan membiarkan *term* tanpa imbuhan *tag*. *Query* yang dimasukkan pengguna dikonversi sesuai aturan tertentu untuk mengekstrak *term-term* penting yang sejalan dengan *term-term* yang sebelumnya telah diekstrak dari dokumen dan menghitung *relevansi* antara *query* dan dokumen berdasarkan pada *term-term* tersebut. Sebagai hasilnya, sistem mengembalikan suatu daftar dokumen terurut *descending* (*ranking*) sesuai nilai kemSistem Temu Kembali Informasi ipannya dengan *query* pengguna (Cios dkk, 2007).



Gambar 2.1 Bagian-bagian Sistem Temu Kembali Informasi

(Sumber : cios, 2007)

2.3 Pembuatan Sistem Temu Kembali Informasi

Menurut Manning (2009) dan terdapat beberapa langkah pembangunan Sistem Temu Kembali Informasi :

1. Mengumpulkan dokumen yang akan di-*index* (dikenal dengan nama *corpus*/koleksi dokumen).
2. Penghapusan format dan *markup* dari dalam dokumen. Pada tahap ini semua *tag markup* dan format khusus dihapus dari dokumen, terutama pada dokumen yang mempunyai banyak *tag* dan format seperti dokumen HTML.
3. Pemisahan rangkaian kata (*tokenization*).

Pada tahapan ini, seluruh kata di dalam kalimat atau pun paragraf dipisahkan menjadi *token* atau potongan kata tunggal atau *termmed word*. Tahapan ini juga menghilangkan karakter-karakter tertentu seperti tanda baca dan mengubah semua *token* ke bentuk huruf kecil (*lowercase*).

4. Penghapusan *stop-words*. *Stop-words* didefinisikan sebagai *term* yang tidak berhubungan (*Sistem Temu Kembali Informasi relevant*) dengan subyek utama dari database meskipun kata tersebut seringkali Sistem Temu Kembali Informasi di dalam dokumen. Kata-kata tersebut termasuk kata penghubung, kata depan, dan sejenisnya. Contoh *stop-words* adalah ini, itu, dia, kami, pada, juga, jika, karena, meskipun, dan sebagainya.
5. *Indexing* (pengindeksan), membangun basis data indeks dari koleksi dokumen. Dilakukan terlebih dahulu sebelum pencarian dokumen dilakukan. Untuk menghasilkan pengindeksan dibutuhkan *stoplist*, *stoplist* adalah kata-kata buang yang tidak akan digunakan sebagai istilah indeks.
6. Pemberian bobot terhadap *term*
Setiap *term* diberikan bobot sesuai dengan skema pembobotan yang dipilih, pada pembobotan BIM digunakan pembobotan biner, pembobotan ini adalah aturan dari metode BIM Sistem Temu Kembali Informasi , dengan melakukan

pengecekan *terms* pada setiap dokumen dan diberi nilai 1 jika ada dan 0 jika tidak ada.

7. Setelah sistem temu kembali informasi menerima *query* dari pengguna, kemudian melakukan perankingan terhadap dokumen pada koleksi berdasarkan kesesuaiannya dengan *query*. Hasil perankingan yang diberikan kepada pengguna merupakan dokumen yang menurut sistem relevan dengan *query*. Namun relevansi dokumen terhadap suatu *query* merupakan penilaian pengguna yang subjektif dan dipengaruhi banyak faktor seperti topik, waktu, sumber informasi maupun tujuan pengguna.

Model sistem temu kembali informasi menentukan detail sistem temu kembali informasi yaitu meliputi representasi dokumen maupun *query*, fungsi pencarian (*retrieval function*) dan notasi kesesuaian (*relevance notation*) dokumen terhadap *query*.

2.3.1 Stemming

Stemming adalah salah satu cara yang digunakan untuk meningkatkan performa sistem temu balik informasi dengan cara mentransformasi kata-kata dalam sebuah dokumen teks ke bentuk kata dasarnya. Tidak banyak algoritma yang dikhususkan untuk *stemming* bahasa Indonesia dengan berbagai keterbatasan didalamnya, diantaranya adalah Algoritma Porter dan Algoritma Nazief & Adriani.

2.3.1.1 Algoritma Porter

Stemming khusus bahasa Inggris yang ditemukan oleh Martin Porter 1980. Mekanisme algoritma dalam mencari kata dasar suatu kata berimbuhan dengan membuang imbuhan– imbuhan (atau lebih tepatnya akhiran) pada kata–kata bahasa Inggris karena dalam bahasa Inggris tidak mengenal awalan. Karena bahasa Inggris datang dari kelas yang berbeda, beberapa modifikasi telah dilakukan untuk membuat Algoritma *Porter* sehingga dapat digunakan sesuai dengan bahasa Indonesia.

2.3.1.1.1 Porter Stemmer Bahasa Indonesia

Implementasi *Porter Stemmer for Bahasa Indonesia* berdasarkan English *Porter Stemmer* yang dikembangkan oleh W.B. Frakes pada tahun 1992. Karena bahasa Inggris datang dari kelas yang berbeda, beberapa modifikasi telah dilakukan untuk membuat Algoritma *Porter* dapat digunakan sesuai dengan bahasa Indonesia.

Algoritma porter yang dibuat oleh W.B Frakes memiliki tahapan-tahapan sebagai berikut (Agusta L,2009) :

1. Hapus *Particle*,
2. Hapus *Possesive Pronoun*.
3. Hapus awalan pertama. Jika tidak ada lanjutkan ke langkah 4a, jika ada cari maka lanjutkan ke langkah 4b.
4. a. Hapus awalan kedua, lanjutkan ke langkah 5a.
b. Hapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai *root word*. Jika ditemukan maka lanjutkan ke langkah 5b.
5. a. Hapus akhiran. Kemudian kata akhir diasumsikan sebagai *root word*
b. Hapus awalan kedua. Kemudian kata akhir diasumsikan sebagai *root word*.

Terdapat 5 kelompok aturan pada Algoritma Porter untuk Bahasa Indonesia ini. Aturan tersebut dapat dilihat pada Tabel 2.1 sampai Tabel 2.5.

Tabel 2.1 aturan untuk *inflectional particle*

Akhiran	Replacement	Measure Condition	Additional Condition	Contoh
-kah	NULL	2	NULL	bukukah
-lah	NULL	2	NULL	pergilah
-pun	NULL	2	NULL	bukupun

Tabel 2.2 Aturan Untuk *Inflectional Possesive Pronoun*

Akhiran	Replacement	Measure Condition	Additional Condition	Contoh
-ku	NULL	2	NULL	bukuku
-mu	NULL	2	NULL	bukumu
-nya	NULL	2	NULL	bukunya

Tabel 2.3. Aturan Untuk *First Order Derivational Prefix*

Awalan	Replacement	Measure Condition	Additional Condition	Contoh
meng-	NULL	2	NULL	mengukur → ukur
meny-	S	2	V...*	menyapu → sapu
men-	NULL	2	NULL	menduga → duga
mem-	P	2	V...	memaksa → paksa
mem-	NULL	2	NULL	membaca → baca
me-	NULL	2	NULL	merusak → rusak
peng-	NULL	2	NULL	pengukur → ukur
peny-	S	2	V...	penyapu → sapu
pen-	NULL	2	NULL	penduga → duga
pem-	P	2	V...	pemaksa → paksa
pem-	NULL	2	NULL	pembaca → baca
di-	NULL	2	NULL	diukur → ukur
ter-	NULL	2	NULL	tersapu → sapu
ke-	NULL	2	NULL	kekasih → kasih

Tabel 2.4. Aturan Untuk *Second Order Derivational Prefix*

Awalan	Replacement	Measure Condition	Additional Condition	Contoh
ber-	NULL	2	NULL	berlari → lari
bel-	NULL	2	Ajar	belajar → ajar
be-	NULL	2	k*er	bekerja → kerja
per-	NULL	2	NULL	perjelas → jelas
pel-	NULL	2	Ajar	pelajar → ajar
pe-	NULL	2	NULL	pekerja → kerja

Tabel 2.5 . Aturan Untuk *Derivational Suffix*

Akhiran	Replacement	Measure Condition	Additional Condition	Contoh
-kan	NULL	2	Prefix bukan anggota {ke, peng}	tarikkan → tarik, mengambilkan → ambil
-an	NULL	2	prefix bukan anggota {di, meng, ter}	makanan → makan, perjanjian → janji
-i	NULL	2	prefix bukan anggota {ber, ke, peng}	Tandai → tanda, mendapati → dapat

2.3.1.2 Algoritma Nazief & Adriani

Algoritma Nazief & Adriani memperhatikan kemungkinan adanya partikel-partikel yang mungkin mengikuti suatu kata berimbuhan. Sehingga kita dapat melihat pada rumus untuk algoritma ini yaitu adanya penempatan *possesive pronoun* dan juga partikel yang mungkin ada pada suatu kata berimbuhan (Agusta, 2009).

Algoritma Nazief & Adriani yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut (Agusta, L.2009):

1. Pertama cari kata yang akan diistem dalam kamus kata dasar. Jika ditemukan maka diasumsikan kata adalah *root word*. Maka algoritma berhenti.
2. *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”) dibuang. Jika berupa *particles* (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada.
3. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah 3a.
 - a) Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “- k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
 - b) Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4. Hilangkan *derivation prefixes DP* {“di-”, “ke-”, “se-”, “me-”, “be ”, “pe”, “te-”} dengan iterasi maksimum adalah 3 kali.
 - a) Langkah 4 berhenti jika:
 1. Terjadi kombinasi awalan dan akhiran.
 2. Awalan yang dideteksi saat ini sama dengan awalan yang dihilangkan sebelumnya.
 3. Tiga awalan telah dihilangkan.
 - b) Identifikasikan tipe awalan dan hilangkan. Awalan ada tipe:
 1. Standar: “di-”, “ke-”, “se-” yang dapat langsung dihilangkan dari kata.
 2. Kompleks: “me-”, “be-”, “pe”, “te-” adalah tipe-tipe awalan yang dapat bermorfologi sesuai kata dasar yang mengikutinya. Oleh

karena itu, gunakan aturan pada Tabel E.1 untuk mendapatkan pemenggalan yang tepat. Tabel dapat dilihat pada LAMPIRAN E.

- c) Cari kata yang telah dihilangkan awalnya ini di dalam kamus. Apabila tidak ditemukan, maka langkah 4 diulangi kembali. Apabila ditemukan, maka keseluruhan proses dihentikan.

5. Apabila setelah langkah 4 kata dasar masih belum ditemukan, maka proses *recoding* dilakukan dengan mengacu pada aturan pada Tabel 2.2 *Recoding* dilakukan dengan menambahkan karakter *recoding* di awal kata yang dipenggal. karakter *recoding* adalah huruf kecil setelah tanda hubung ('-') Dan terkadang berada sebelum tanda kurung. Sebagai contoh, kata “menangkap” (aturan 15), setelah dipenggal menjadi “nangkap”. Karena tidak valid, maka *recoding* dilakukan dan menghasilkan kata “tangkap”.
6. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata Awal diasumsikan sebagai *root word*. Proses selesai.

Tipe awalan ditentukan melalui langkah-langkah berikut:

1. Jika awalnya adalah: “di-”, “ke-”, atau “se-” maka tipe awalnya secara berturut-turut adalah “di-”, “ke-”, atau “se-”.
2. Jika awalnya adalah “te-”, “me-”, “be-”, atau “pe-” maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalnya.
3. Jika dua karakter pertama bukan “di-”, “ke-”, “se-”, “te-”, “be-”, “me-”, atau “pe-” maka berhenti.
4. Jika tipe awalan adalah “none” maka berhenti. Jika tipe awalan adalah bukan “none” maka awalan dapat dilihat pada Tabel 3. Hapus awalan jika ditemukan.

Untuk mengatasi keterbatasan pada algoritma diatas, maka ditambahkan aturan-aturan dibawah ini :

1. Aturan untuk reduplikasi.

a) Jika kedua kata yang dihubungkan oleh kata penghubung adalah kata yang sama maka *root word* adalah bentuk tunggalnya, contoh : “buku-buku” *root word*-nya adalah “buku”.

b) Kata lain, misalnya “bolak-balik”, “berbalas-balasan, dan ”seolah-olah”. Untuk mendapatkan *root word*-nya, kedua kata diartikan secara terpisah. Jika keduanya memiliki *root word* yang sama maka diubah menjadi bentuk tunggal, contoh: kata “berbalas-balasan”, “berbalas” dan “balasan” memiliki *root word* yang sama yaitu “balas”, maka *root word* “berbalas-balasan” adalah “balas”. Sebaliknya, pada kata “bolak-balik”, “bolak” dan “balik” memiliki *root word* yang berbeda, maka *root word*-nya adalah “bolak-balik”.

2. Tambahan bentuk awalan dan akhiran serta aturannya. Untuk tipe awalan “mem-“, kata yang diawali dengan awalan “memp”.

Information retrieval memiliki 3 model yang dapat digunakan, yaitu model boolean, model probabilistic, dan model ruang vektor.

2.4 Model Probabilistik

Model probabilistik adalah model sistem temu kembali informasi yang mengurutkan dokumen dalam urutan menurun terhadap peluang relevansi sebuah dokumen terhadap informasi yang dibutuhkan. Beberapa model yang juga dikembangkan berdasarkan perhitungan probabilistik yaitu, *Binary Independence Model*, model Okapi BM25, dan *Bayesian Network Model* (Manning dkk, 2009).

Dalam model probabilistik dasar, kemSistem Temu Kembali Informasi ipan (*similarity*) sebuah dokumen terhadap query dihitung dengan menggunakan rumus seperti pada Persamaan 2.5.

$$RSV(d) = \sum_{t \in q} \left\{ \log \left(\frac{(s + 0,5) / (S - s + 0,5)}{(df(t) - s + 0,5) / (N - df(t) - S + s + 0,5)} \right) \right\} \dots\dots\dots(2.1)$$

Keterangan:

RSV = *retrieval status value* (nilai untuk perangkingan dokumen).

s = jumlah dokumen yang relevan yang mengandung *termt* pada query *q*.

S = jumlah dokumen yang relevan untuk query *q*.

df_t = jumlah dokumen dalam *corpus* yang mengandung *termt* pada query *q*.

N = jumlah dokumen dalam *corpus*.

2.4.1 *Binary Independence Model (BIM)*

Pada model *Binary Independence Model*, query dianggap sebagai sebuah vector term. Jika pada model lain jumlah atau kemunculan term diperhitungkan, maka pada Model ini nilainya berupa biner, Yaitu ada atau tidak ada.

Rumus relevansi *Binary Independence Model* :

$$Sim(Q,D) = \sum_{k=1}^n w_{ki} \cdot w_{kj} \log \frac{p_k (1-q_k)}{q_k (1-p_k)} \dots\dots\dots(2.2)$$

k = kata yang ada dalam query

n = jumlah kata pada query

w_k = bobot biner dari kata tersebut terhadap dokumen query

p_k = peluang sebuah dokumen yang relevan mengandung kata pada query k

q_k = peluang sebuah dokumen yang tidak relevan mengandung kata pada query k

pada inisialisasi nilai p_k ini tidak diketahui dan biasanya diisi dengan angka 0,5 (Taufik Ramadhany,2008). Sedangkan q_k didapatkan dari perhitungan *ni/N*, dengan ni adalah jumlah dokumen yang mengandung term i, sedangkan N adalah jumlah total dokumen dalam koleksi.

2.5 Model Boolean

Dengan model boolean maka pencarian *query* dilakukan dengan fungsi-fungsi logika yang umum seperti OR, AND, XOR, NOT, NAND, NOR dan lain sebagainya diantara kata yang diinginkan. Contohnya jika query $Q = (K1 \text{ AND } K2) \text{ OR } (K3 \text{ AND } (\text{NOT } K4))$.

Pencarian Boolean akan mengambil semua dokumen yang di indeks oleh K1 dan K2, seperti halnya semua dokumen yang diindeks oleh K3 yang tidak termasuk dalam indeks K4. Suatu cara mengimplementasikan pencarian Boolean adalah melalui *inverted file*. Kita menyimpan daftar tiap kata kunci dalam sebuah kata dan tiap kata menunjuk alamat dokumen yang mengandung kata tertentu itu untuk memaksimalkan *query* kita melakukan set terhadap operasi dan menghubungkannya dengan daftar koleksi (*K-List*). Sebagai contoh jika:

K1 - List: D1, D2, D3, D4

K2 - List: D1, D2

K3 - List: D1, D2, D3

K4 - List: D1

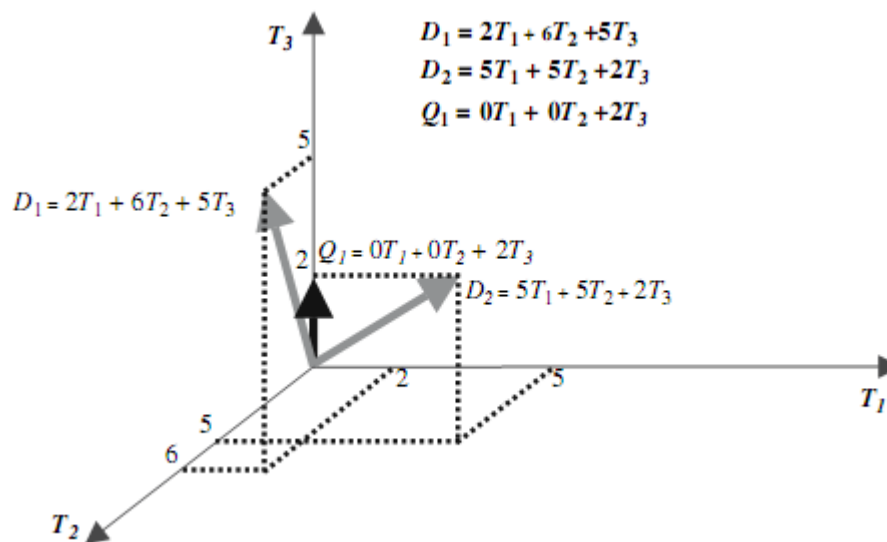
Dan $Q = (K1 \text{ AND } K2) \text{ OR } (K3 \text{ AND } (\text{NOT } K4))$

Untuk memaksimalkan bagian ($K1 \text{ AND } K2$) kita lakukan *intersect* (Sistem Temu Kembali Informasi) terhadap *K1 - List* dan *K2 - List*. dan untuk memaksimalkan ($K3 \text{ AND } (\text{NOT } K4)$) maka kita mengurangi *K4 - List* dari *K3 - List*. OR di maksimalkan dengan mengambil *union* (gabungan) dari dua set dokumen yang diperoleh dari masing-masing bagian tadi. Hasil akhir Sistem Temu Kembali Informasi nya berupa {D1, D2, D3} yang merupakan *query* maksimal dan tiap dokumen didalamnya benar untuk *query* tersebut.

Dengan menggunakan *operator* logika dalam boolean memiliki kelebihan dan kelemahan tersendiri Sistem Temu Kembali Informasi. Kelebihannya adalah lebih mudah bagi *user* yang berpengalaman dan kelemahannya adalah kerumitan dalam penggunaan bahasa *query* dan akan membingungkan pengguna yang biasa.

2.6 Model Ruang Vektor (Vector Space Model – VSM)

Dalam sistem IR, kemiripan antar dokumen didefinisikan berdasarkan representasi *bag of words* dan dikonversikan ke suatu model ruang vektor (*vector space model* - VSM). Pada VSM, setiap dokumen di dalam *database* dan *query* pengguna direpresentasikan oleh suatu vektor multi-dimensi seperti yang ditunjukkan oleh Gambar 2.2.



Gambar 2.2. Contoh VSM dengan dua dokumen D_1 dan D_2 , dan *query* Q_1
(Sumber: Cios dkk, 2007)

Berdasarkan Gambar 2.2, dapat diketahui bahwa sudut yang dibentuk antara Q_1 dan D_1 lebih kecil daripada Q_1 dan D_2 . Perhitungan persamaan antara vektor *query* dan vektor dokumen dilihat dari sudut yang terkecil, yaitu antara Q_1 dan D_1 . Sudut yang dibentuk oleh dua vektor ini dapat dihitung dengan melakukan perkalian dalam (*inner product*), sehingga rumus relevansinya adalah:

$$R(Q, D) = \cos \theta = \frac{Q \cdot D}{|Q||D|} \dots\dots\dots(2.3)$$

Berbeda dengan model boolean yang menggunakan nilai biner sebagai bobot index

term, VSM melakukan pembobotan berdasarkan *term* yang sering muncul dalam dokumen atau dikenal dengan sebutan *term frequency* (*tf*) dan jumlah kemunculannya dalam koleksi dokumen yang disebut *inverse document frequency* (*idf*) (Manning dkk, 2009).

Pada model ruang vektor, pembobotan terhadap *term* dilakukan dengan mengalikan bobot lokal *tf* dan bobot global *idf*, dikenal dengan pembobotan *tf-idf*. Metode pembobotan ini dilakukan dengan memberikan bobot kepada *term* yang penting. Artinya, *term* yang jika muncul di suatu dokumen maka, dokumen tersebut dapat dianggap relevan dengan *query* pengguna.

$$idf(t) = \log \left(\frac{N}{df(t)} \right) \dots\dots\dots(2.4)$$

$$w = tf(td) \cdot idf(t) \dots\dots\dots(2.5)$$

Keterangan:

N : jumlah dokumen dalam *corpus*.

df_t : *document frequency* atau jumlah dokumen dalam *corpus* yang mengandung *term* t .

w : *weighting* (bobot).

tf_{td} : *term frequency* atau jumlah kemunculan *term* t dalam dokumen d .

idf_t : *inverse document frequency*, berbanding terbalik dengan df_t .

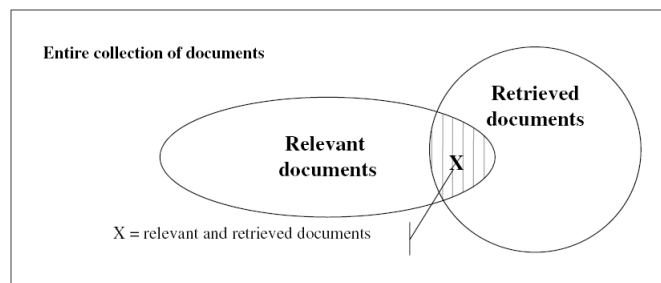
Kemiripan antar dokumen dihitung menggunakan suatu fungsi kemiripan (*similarity measure*). Ukuran ini memungkinkan perbandingan dokumen sesuai dengan kemiripannya (relevansinya) terhadap *query*. Setelah dokumen di-*ranking*, sejumlah dokumen *top-scoring* dikembalikan ke *user*. Salah satu ukuran kemiripan teks yang populer adalah *cosine similarity*, yang menghitung nilai cosinus sudut antara dua vektor, dimana perhitungan ini didasarkan pada Persamaan 2.1. Misalkan terdapat dua vektor dokumen d_j dan $query_q$, serta $tterm$ yang diekstrak dari koleksi dokumen maka, nilai cosinus antara d_j dan q didefinisikan sebagai Persamaan 2.4 (Cios dkk, 2007).

$$\text{similarity}(d(j), q) = \frac{d(j) \cdot q}{|d(j)| \cdot |q|} = \frac{\sum_{i=1}^t (w(ij) \cdot w(iq))}{\sqrt{\sum_{i=1}^t w(ij)^2} \cdot \sqrt{\sum_{i=1}^t w(iq)^2}} \dots\dots\dots(2.6)$$

2.7 Recall dan Precision

Ada beberapa alasan yang berbeda mengapa tahap evaluasi Sistem Temu Kembali Informasi (*Information Retrieval*) adalah sesuatu yang penting. Sebagai contoh, penyedia sumber informasi membutuhkan informasi tentang penggunaan sumber daya oleh *user*, dan organisasi yang bekerja untuk meningkatkan kinerja *search engine* perlu metode-metode yang efektif untuk mengevaluasi perubahan yang dilakukan untuk algoritma dan *user interface*. Dengan demikian, tujuan evaluasi adalah untuk menghasilkan perbaikan pada proses pengambilan informasi. Di sisi lain, tujuan evaluasi biasanya tergantung pada penelitian, beberapa peneliti mungkin berpendapat bahwa tujuan utama dari evaluasi adalah untuk mengevaluasi kekuatan metodologi pengindeksan dan pencarian, namun beberapa fokus lain dari penelitian evaluasi *information retrieval* adalah proses kognitif pengguna, antarmuka manusia-komputer, dan karakteristik *database* (Zhang, 2008).

Information retrieval system mengembalikan satu set dokumen sebagai jawaban atas *user's query*. Terdapat dua kategori dokumen yang dihasilkan oleh sistem Sistem Temu Kembali Informasi terkait pemrosesan *query*, yaitu *relevant document* (dokumen yang relevan dengan *query*) dan *retrieved document* (dokumen yang diterima pengguna). Hubungan antara kedua kategori ini digambarkan menggunakan diagram Venn pada gambar 2.7. (Cios, 2007):



Gambar 2.4 Relasi antara *relevant* dan *retrieve* dokumen

Ukuran umum yang digunakan untuk mengukur kualitas dari *text retrieval* adalah kombinasi *precision* dan *recall*.

Pada dasarnya, nilai *precision* dan *recall* bernilai antara 0-1. Oleh karena itu, suatu sistem Sistem Temu Kembali Informasi yang baik diharapkan untuk dapat memberikan nilai *precision* dan *recall* mendekati 1. *Precision* dan *recall* adalah faktor penting dalam mengevaluasi sistem Sistem Temu Kembali Informasi, Kondisi antara *precision* dan *recall* mengakibatkan terjadi 2 situasi ekstrim berikut (Cios, 2007):

1. *Precision* terlalu tinggi dan *recall* rendah. Sistem mengembalikan beberapa dokumen dan Sistem Temu Kembali Informasi semuanya relevan, tetapi sejumlah besar dokumen relevan lain terabaikan.
2. *Recall* sangat tinggi dan *precision relatif* rendah. Sistem mengembalikan sejumlah besar dokumen yang mengikutsertakan Sistem Temu Kembali Informasi semua dokumen relevan tetapi juga mencakup sebagian besar dokumen yang tak diharapkan.

Pengujian kemampuan sistem *information retrieval* dilakukan dengan menghitung nilai *precision* dan *recall* berdasarkan korelevanan sistem menampilkan dokumen sesuai dengan *query*. Nilai *precision* adalah keakurasian atau kecocokan (antara permintaan informasi dengan jawaban terhadap permintaan itu) jika seseorang mencari informasi di sebuah sistem, dan sistem menawarkan beberapa dokumen maka keakurasian ini sebenarnya juga adalah *relevansi*. Artinya, seberapa persis atau cocok dokumen tersebut untuk keperluan pencari informasi, bergantung kepada seberapa *relevan* dokumen tersebut bagi si pencari. Kemudian *recall* adalah proporsi jumlah dokumen yang dapat ditemu kembalikan oleh sebuah proses pencarian sistem *Information retrieval*. Berikut Tabel 2.1 parameter untuk menghitung *precision* dan *recall*.

Tabel 2.1 Parameter Menghitung Precision dan *Recall*.

keterangan	<i>relevan</i>	Tidak <i>relevan</i>
terambil	<i>True positive</i> (tp)	False positive (fp)
Tidak terambil	false negative (fn)	True negative (tn)

Rumus untuk menghitung *Precision*:

$$P = tp / (tp + fp) \dots \dots \dots (2.7)$$

Keterangan :

$P = Precision$

$tp = true\ positive$

$fp = false\ positive$

Rumus untuk menghitung *Recall*:

$$R = tp / (tp + fn) \dots \dots \dots (2.8)$$

Keterangan :

$R = Recall$

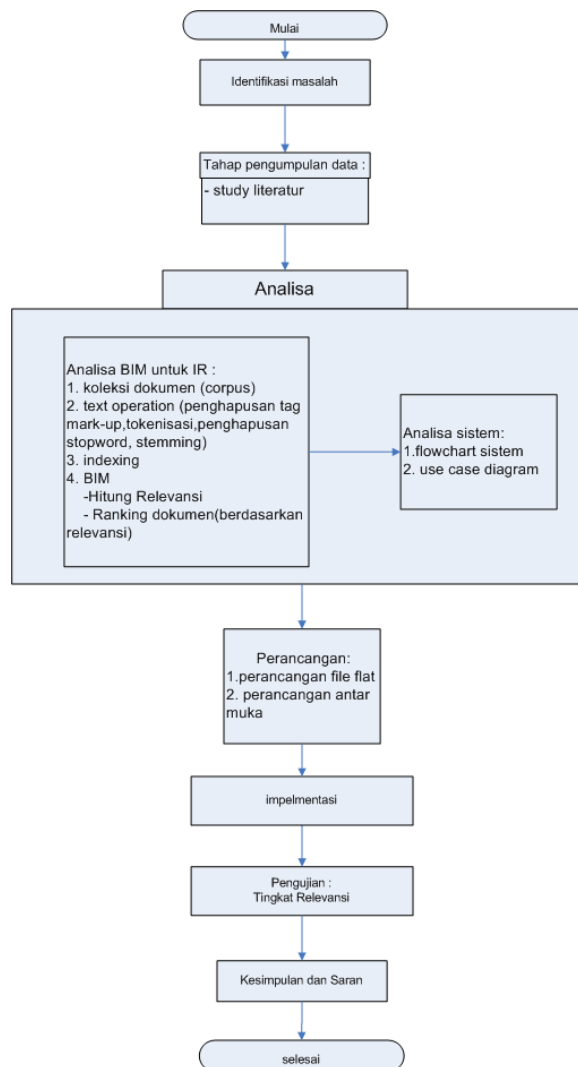
$Tp = true\ positive$

$Fn = false\ negative$

BAB III

METODOLOGI PENELITIAN

Metodologi penelitian menjelaskan bagaimana langkah-langkah atau tahapan-tahapan yang akan dilakukan dalam penelitian untuk dapat menjawab perumusan masalah penelitian.



Gambar 3.1 Tahapan Penelitian

3.1. Identifikasi Masalah

Pada tahapan ini dilakukan identifikasi permasalahan bahwa pentingnya bagi *user* untuk memperoleh informasi yang relevan sesuai dengan kebutuhannya (yang diekspresikan melalui *query*) dari sekumpulan informasi (dokumen).

3.2. Tahap pengumpulan data

Pada tahapan ini akan dijelaskan tentang tahap-tahap pengumpulan data dalam penelitian yang akan dilakukan. Terdapat dua tahap dalam pengumpulan data yang dilakukan :

1. Studi Literatur. Melakukan pengumpulan informasi melalui jurnal-jurnal ilmiah dan buku-buku yang berhubungan dengan permasalahan pada penelitian tugas akhir ini. Sehingga memperoleh referensi untuk dapat menerapkannya pada tugas akhir ini dan dapat menyelesaikan masalah-masalah saat melakukan penelitian.

3.3 Analisa

Pada tahapan ini dijelaskan tentang analisa sistem dan permasalahan yang akan di bangun sehingga memperjelas tentang pemahaman sistem. Hal-hal yang akan dianalisa adalah :

3.3.1 Analisa BIM untuk Sistem temu kembali informasi

Pada tahapan ini akan dijelaskan secara rinci tentang analisa metode BIM Pada sistem temu kembali informasi yang akan dibangun sehingga mempermudah pemahaman terhadap sistem:

1. Koleksi dokumen (*corpus*), mengumpulkan dokumen untuk kepentingan penelitian
2. *Text operation*, melakukan penghapusan pada tag-tag pada .html, memecah kata-kata pada dokumen yang dikoleksi (*tokenization*), penghapusan kata-kata

yang tidak mempunyai makna (*stopword*), menghapuskan kata-kata yang mempunyai imbuhan dan menyisakan kata dasar (*Stemming*)

3. *Indexing*, memberi tanda untuk setiap kata yang telah di *tokenization*.
4. BIM (*Binary Independence Model*)
 - Hitung *relevansi* dokumen terhadap *query* dan dokumen
 - Melakukan perangkingan dokumen dari *query* yang diberi nilai *relevansi*

3.3.2 Analisa Sistem

1. *Flowchart* sistem, membuat analisa aliran sistem.
2. *Usecase* diagram, menganalisa hubungan antara pengguna dengan sistem.

3.4 Perancangan

1. Perancangan *file flat*, misalnya seperti *token.txt*, *stopword.txt*
2. Perancangan antar muka, merancang antar muka yang dipahami oleh pengguna.

3.5 Implementasi Sistem

Pada tahapan implementasi ini akan dilakukan pembuatan modul-modul yang telah dirancang dalam tahap perancangan kedalam bahasa pemrograman. Implementasi sistem akan dilakukan dengan spesifikasi sebagai berikut :

Perangkat Keras

Processor : *Pentium® dual-core T2390, 1,86GHz*

Memori (RAM) : 1.5 GB

Perangkat Lunak

Sistem Operasi : *Windows 7 Professional*

Bahasa Pemrograman : PHP

Tools Perancang : XAMPP 1.6.6a

Web Browser : *Firefox*

3.6 Pengujian Tingkat Relevansi

Pengujian merupakan tahapan dimana sistem akan dijalankan. Tahap pengujian diperlukan untuk menjadi ukuran bahwa sistem dapat dijalankan sesuai dengan tujuan. Pengujian sistem IR dalam tugas akhir ini dilakukan dengan cara mengukur kualitas *text retrieval*. Ukuran yang digunakan untuk mengukur kualitas dari *text retrieval* adalah *precision* dan *recall*. Serta dilakukan penarikan kesimpulan terhadap hasil pengujian sistem.

3.7 Kesimpulan dan Saran

Dalam tahapan ini dilakukan penarikan kesimpulan terhadap hasil penelitian yang telah dilakukan untuk mengetahui apakah implementasi sistem yang telah dilakukan dapat beroperasi dengan baik dan sesuai dengan tujuan yang diinginkan, serta memberikan saran-saran untuk pengembangan penelitian selanjutnya agar tercipta suatu rancangan mesin IR yang sempurna.

BAB IV

ANALISA DAN PERANCANGAN

Bab ini berisi analisa pembahasan mengenai analisa dan perancangan alur operasi *Binary Independence Model* pada *information retrieval* terhadap ekspresi kebutuhan pengguna (*query*) untuk menguji representasi *Binary Independence Model* terhadap *information retrieval*.

4.1 Analisa *Binary Independence Model* (BIM) untuk IR

Terdapat beberapa tahapan penerapan model BIM pada IR, diantaranya :

- a. Koleksi dokumen (*corpus*) yang akan di *index*.
- b. Tahap penghapusan *tag markup* dan format khusus dari dalam dokumen, tokenisasi, tahap penghapusan *stop-words* dan tahap penerapan *stemming*. Keseluruhan tahap ini selanjutnya akan disebut *text operation* dalam bab ini.
- c. *Indexing* (pembuatan *inverted index*).
- d. Pembobotan (*weighting*) terhadap kata (*term*) hasil *indexing*.
- e. Menghitung nilai relevansi antara *query* dan dokumen.
- f. Melakukan perankingan dokumen dari nilai relevansi yang diperoleh.

4.1.1 Koleksi Dokumen (*Corpus*)

Dokumen merupakan objek utama dalam penelitian tugas akhir ini sebagai sumber informasi yang akan di-*retrieved* melalui sistem temu kembali informasi dengan *Binary Independence Model* sesuai dengan *query* yang diinputkan oleh pengguna. Dalam penelitian tugas akhir ini akan digunakan *corpus*. *Corpus* atau disebut juga koleksi dokumen merupakan kumpulan besar sejumlah dokumen atau teks yang digunakan untuk kajian atau penelitian linguistik. *Corpus* yang digunakan dalam penelitian tugas akhir ini adalah *corpus* yang juga digunakan untuk bahan eksperimen dalam penelitian dibidang IR (*Information Retrieval*) berbahasa Indonesia. Jumlah dokumen yang dikumpulkan dalam penelitian ini berjumlah 281 dokumen yang berformat .html,htm, .txt dan pdf.

4.1.2 Text Operation terhadap koleksi dokumen

Pada tahapan ini dilakukan pemecahan kata terhadap teks yang terdapat pada *corpus* untuk menghasilkan kosa kata. Untuk memperoleh kata tunggal (*termmed word*), yang selanjutnya akan digunakan dalam proses *indexing*, maka terlebih dahulu akan dilakukan tahapan penghapusan *tag markup* dan format khusus dari dalam dokumen, kemudian melakukan *tokenisasi* (pemisahan rangkaian kata), dan melakukan *linguistic preprocessing* (penghapusan *stop-words*) untuk setiap dokumen di dalam *corpus*.

1. Penghapusan *tag markup* dan format khusus dalam dokumen.

Sebelum melakukan *tokenisasi* (pemecahan kata), semua *tag mark up* dan format khusus akan dihapus dari dalam dokumen, terutama pada dokumen yang berformat .htm dan .html yang memiliki banyak *tag*, contoh tag `<ahref>`, ``, ``, ``, `<p>` `</p>` dan *tag* lainnya serta *javascript* dan *cascading style sheet (css)*.

2. Tokenisasi yaitu proses pemisahan kata.

Setelah dilakukannya penghapusan *tag markup*, dilakukan pemisahan kata atau *Tokenisasi*. *Tokenisasi* yang dilakukan akan menghasilkan potongan kata

tunggal (*term*) yang nantinya akan diindeks. Dalam proses ini juga dilakukan penghapusan karakter-karakter tertentu, yaitu tanda baca serta mengubah semua kata (*term*) ke bentuk huruf kecil (*lowercase*).

Contoh :

Dok 1 : Bank Mandiri mengucurkan pembiayaan.

Table 4.1 Hasil tokenisasi

bank
mandiri
mengucurkan
pembiayaan

3. *Linguistic preprocessing.*

Penghapusan *stop-words*. dimana term/kata-kata yang dianggap umum akan diabaikan dalam proses *indexing*, Karena itu, *term* tersebut dihapus dari dalam dokumen. Dalam penelitian ini, daftar *stop-words* ditentukan sebelumnya, kemudian disimpan dalam pembangun sistem untuk pemrosesan sistem temu kembali informasi dengan *Binary Independence model*.

4. *stopword* adalah kata-kata buang yang tidak akan digunakan sebagai istilah indeks. *Stopword* sangat diperlukan dalam sistem temu kembali informasi karena kata-kata penghubung yang diinputkan pada *query* dianggap tidak perlu karena hanya akan memperlambat kinerja sistem. Berikut adalah daftar *stopword*.

Contoh kasus :

Terdapat 6 (enam) dokumen di dalam *corpus*. Masing-masing dokumen diberi nama dok 1.txt, dok 2.txt dan dok 3.txt.

Dok 1 : Indonesia memiliki kesempatan menyalip posisi Inggris, karena sepanjang tahun ini pertumbuhan pengguna Twitter di Indonesia lebih besar dibanding negara penyelenggara Olimpiade 2012 ini.

Dok 2 : PBB mengonfirmasi pada Rabu bahwa pemberontak yang memerangi rezim Presiden Bashar al-Assad sekarang memiliki senjata berat, dan pengamat militer melihat penggunaan jet tempur militer Suriah untuk menyerang pemberontak di Aleppo.

Dok 3 : Bank Mandiri sebelumnya juga telah mengucurkan pembiayaan untuk membantu pengembangan perusahaan anak Semen Gresik antara lain Semen Tonasa, Varia Usaha, Swadaya Graha, Koperasi Warga Semen Gresik dan perusahaan anak lainnya.

Dok 4 : Hans Prawira terlihat cukup lihai mengucapkan salam kepada pelanggan dan mengoperasikan komputer kasir. Bahkan ia pun mencoba menawarkan layanan lain dan program-program yang berlaku di toko tersebut.

Dok 5 : Aparat Subdit Cyber Crime Polda Metro Jaya membongkar praktik ilegal penjualan nomor rekening bank melalui situs internet. Sejumlah tersangka diamankan dalam kasus tersebut.

Dok 6 : Dengan menawarkan layanan streaming yang bisa disesuaikan dengan selera dan kebutuhan pengguna, Apple bisa bersaing dengan layanan internet radio seperti Pandora, Slacker dan iHeartRadio.

Untuk contoh kasus ini, misalkan daftar *stop-words* yang digunakan, yaitu :

adalah, adanya, antara, apa, atau, bahwa, berbagai, dalam, dan, dari, dengan, di, ini, itu, jika, juga, karena, ke, lalu, merupakan, meski, meskipun, oleh, pada, padahal, para, sedemikian, serta, telah, tentang, tersebut, tidak, untuk, yaitu, yang, dia, kami, kamu, mereka, saya.

Table 4.2 Hasil *text operations* terhadap dokumen contoh kasus

<i>Dok 1</i>	<i>Dok 2</i>	<i>Dok 3</i>	<i>Dok 4</i>	<i>Dok 5</i>	<i>Dok 6</i>
Indonesia	PBB	Bank	hans	aparatus	menawarkan
memiliki	Mengonfirmasi	Mandiri	prawira	subdit	layanan
kesempatan	Rabu	mengucurkan	terlihat	cyber	streaming
menyalip	pemberontak	pembiayaan	cukup	crime	bisa
posisi	memerangi	membantu	lihai	polda	disesuaikan
inggris	Rezim	pengembangan	mengucapkan	metro	selera
sepanjang	Presiden	perusahaan	salam	jaya	kebutuhan
tahun	Bashar al-Assad	Anak	kepada	membongkar	pengguna
pertumbuhan	memiliki	Semen	pelanggan	praktik	Apple
pengguna	Senjata	Gresik	mengoperasikan	ilegal	bisa
Twitter	Berat	Semen	komputer	penjualan	bersaing
Indonesia	pengamat	Tonasa	kasir	nomor	internet
besar	Militer	Varia	mencoba	rekening	radio
banding	Melihat	Usaha	menawarkan	bank	Pandora
negara	penggunaan	Swadaya	layanan	melalui	Slacker
penyelenggara	Jet	Graha	lain	situs	iHeartRadio
Olimpiade	Tempur	Koperasi	program	internet	
2012	Militer	Warga	berlaku	sejumlah	
-	Suriah	Semen	toko	tersangka	
-	menyerang	Gresik	tersebut	amankan	
-	pemberontak	perusahaan	-	kasus	
-	Aleppo	Anak	-	tersebut	

5. *Stemming*

Pada tahapan ini dilakukan *stemming*, menghilangkan imbuhan pada kata yang berimbuhan. Pembahasan *stemming* telah dijelaskan pada BAB II.

Contoh penggunaan *stemming* pada penelitian ini adalah :

Ketika *query* yang dimasukkan adalah “memakan buah apel” maka kata yang berimbuhan “memakan” akan menjadi “makan” karena imbuhan me- dan akhiran -an akan dihilangkan sesuai dengan aturan pada algoritma *stemming*. Algoritma *stemming* yang digunakan dalam penelitian ini adalah algoritma Nazief & Adriani karena memiliki kemampuan presentase keakuratan (presisi) lebih baik dari algoritma porter.

4.1.3 *Indexing*

Dalam tahapan ini, dilakukan proses pengindeksan dokumen terhadap setiap kosa kata yang muncul (hasil tahapan *text operation*) dengan membuat *inverted index*. Setiap kosa kata didaftarkan (*dictionary*) dan didefinisikan di dokumen mana kata tersebut muncul (*postings*). Dalam tugas akhir ini, selain menyimpan daftar informasi id dok, masing-masing *postings* juga akan menyimpan informasi berupa *term frequency (tf)* atau frekuensi kemunculan *term* di dalam dokumen terkait.

Dari daftar informasi id dok yang tersimpan dalam *postings* terhadap masing-masing *term*, didapatkan informasi mengenai *document frequency(df)* atau jumlah dokumen yang mengandung *term* tersebut.

Tabel 4.3. Struktur *dictionary* dan *postings* (id dok:tf) hasil *indexing* terhadap dokumen contoh kasus

<i>Dictionary</i>	<i>Doc :TF</i>
Indonesia	1:1
memiliki	1:1

Pada *Doc : TF* terdapat angka 1:1 maksudnya adalah dokumen terletak pada dokumen 1 setelah tanda (:) penjelasannya adalah jumlah dokumen yang muncul pada dokumen sebanyak 1 kali. Table lebih lengkap terdapat di Lampiran C.

4.1.4 Pembobotan Kata (*Term*) Hasil *Indexing*

Seluruh kosa kata yang telah diperoleh dari hasil proses *indexing* diberikan nilai bobot, pada analisa tugas akhir ini, pembobotan yang dilakukan adalah memberikan nilai 1 dan 0 pada masing-masing term pada *query* dan juga pada koleksi dokumen. pada table pembobotan biner, terdapat $W(n)$ yaitu bobot dokumen ke n , Df adalah *dokumen frekuensi* atau jumlah dokumen yang mengandung *query k*, sedangkan q_k adalah peluang sebuah dokumen yang tidak relevan mengandung kata pada *query k*. nilai q_k disini didapat berdasarkan rumus 2.2 pada bab II.

Contoh pembobotan kata hasil *indexing*:

Dok 1 : Indonesia memiliki kesempatan menyalip posisi Inggris,

Dok 2: bahwa pemberontak yang memerangi rezim Presiden Bashar al-Assad sekarang memiliki senjata berat

Jumlah dokumen (N) = 2

Query : indonesia memiliki

n_i = indonesia

n_i = memiliki

Table 4.3 Contoh pembobotan

Dictionary	Pembobotan (W)dok 1	Pembobotan (W)dok 2	DF	Qk (ni/N)
indonesia	1	0	1	$1/2 = 0.5$
memiliki	1	1	2	$2/2 = 1$

Pembobotan hasil indeks terhadap contoh kasus pada hal IV-3 dapat dilihat di Lampiran D.

4.1.5 Pembobotan Kata Pada *Query*

Pembobotan kata pada *query* menggunakan pembobotan Biner yaitu 1 dan 0, 1 berarti ada dan 0 berarti tidak. Nilai term dari *query* yang diinputkan oleh pengguna akan diproses untuk menghitung *relevansi* seperti pada persamaan 2.2. dimisalkan *query* yang diinputkan adalah “ teknologi indonesia pada tahun 2012” maka pembobotan pada setiap kata pada *query* yang diinputkan adalah :

$$Tf_{iq}, \text{ kata 1: teknologi} = 1$$

$$Tf_{iq}, \text{ kata 2: Indonesia} = 1$$

$$Tf_{iq}, \text{ kata 3: Tahun} = 1$$

$$Tf_{iq}, \text{ kata 4: 2012} = 1$$

Pembobotan pada *query* setelah *query* diindeks akan diberi nilai 1, karena metode menganggap queri itu “ada”.

Namun, nilai tf pada penelitian ini telah ditentukan sebelumnya yaitu 1. Jika terdapat Tf_{iq} yang ganda akan tetap diberi nilai 1 untuk kemunculannya, karena metode BIM telah menentukan aturan tidak menghitung jumlah kata yang muncul dalam setiap *query* yang diinputkan. Sehingga perhitungan proses relevansi dalam penelitian ini digunakan persamaan 2.2.

4.1.6 Perhitungan Nilai Kerelevanan Dokumen Terhadap *Query* dan Perangkingan Dokumen

Setelah proses (tahap) pada subbab 4.1.1.4 selesai dilakukan, maka tahap selanjutnya adalah penghitungan nilai relevansi dokumen terhadap *query* menggunakan Persamaan 2.2.

Query : indonesia mengkonfirmasi bank mandiri menawarkan layanan streaming

Berdasarkan table 4.2, dapat dilihat bahwa dokumen-dokumen yang mengandung kata-kata didalam *query* “teknologi Indonesia pada tahun 2012” adalah dokumen dengan id dok 1. Sehingga dokumen yang dihitung nilai relevansinya adalah dok 1.

$$SIM(d1) = 1.1 \log \frac{0.5(1-0.1666)}{0.1666(1-0.5)} + SIM(d1) = 1.0 \log 0.5 +$$

$$SIM(d1) = 1.0 \log 0.5 + SIM(d1) = 1.0 \log 0.5 + SIM(d1) = 1.0 \log 0.5 + \\ SIM(d1) = 1.0 \log 0.5 + SIM(d1) = 1.0 \log 0.5$$

$$SIM(d1) = 0.69897$$

$$SIM(d2) = 1.0 \log 0.5 + SIM(d2) = 1.1 \log \frac{0.5(1-0.1666)}{0.1666(1-0.5)} + SIM(d2) = \\ 1.0 \log 0.5 + SIM(d2) = 1.0 \log 0.5 + SIM(d2) = 1.0 \log 0.5 + SIM(d2) = \\ 1.0 \log 0.5 + SIM(d1) = 1.0 \log 0.5$$

$$SIM(d2) = 0.69897$$

$$SIM(d3) = 1.0 \log 0.5 + SIM(d3) = 1.0 \log 0.5 + \\ SIM(d3) = 1.1 \log \frac{0.5(1-0.333)}{0.333(1-0.5)} + SIM(d2) = 1.1 \log \frac{0.5(1-0.1666)}{0.1666(1-0.5)} + \\ SIM(d3) = 1.0 \log 0.5 + SIM(d3) = 1.0 \log 0.5 + SIM(d3) = 1.0 \log 0.5$$

$$SIM(d3) = 0.698 + 0.301$$

$$SIM(d3) = 0.999$$

$$\begin{aligned} \text{SIM}(d4) &= 1.0 \log 0.5 + \text{SIM}(d4) = 1.0 \log 0.5 + \text{SIM}(d4) = 1.0 \log 0.5 + \\ \text{SIM}(d4) &= 1.0 \log 0.5 + \text{SIM}(d4) = 1.1 \log \frac{0.5(1-0.333)}{0.333(1-0.5)} + \text{SIM}(d4) = \\ 1.1 \log \frac{0.5(1-0.333)}{0.333(1-0.5)} + \text{SIM}(d4) &= 1.0 \log 0.5 \end{aligned}$$

$$\text{SIM}(d4) = 0.301 + 0.301$$

$$\text{SIM}(d4) = 0.602$$

$$\begin{aligned} \text{SIM}(d5) &= 1.0 \log 0.5 + \text{SIM}(d5) = 1.0 \log 0.5 + \\ \text{SIM}(d5) &= 1.1 \log \frac{0.5(1-0.333)}{0.333(1-0.5)} + \text{SIM}(d5) = 1.0 \log 0.5 + \text{SIM}(d5) = \\ 1.0 \log 0.5 + \text{SIM}(d5) &= 1.0 \log 0.5 + \text{SIM}(d5) = 1.0 \log 0.5 \\ \text{SIM}(d5) &= 0.301 \end{aligned}$$

$$\begin{aligned} \text{SIM}(d6) &= 1.0 \log 0.5 + \text{SIM}(d6) = 1.0 \log 0.5 + \text{SIM}(d6) = 1.0 \log 0.5 \\ + \text{SIM}(d6) &= 1.0 \log 0.5 + \text{SIM}(d6) = 1.1 \log \frac{0.5(1-0.333)}{0.333(1-0.5)} + \text{SIM}(d6) = \\ 1.1 \log \frac{0.5(1-0.333)}{0.333(1-0.5)} + \text{SIM}(d6) &= 1.1 \log \frac{0.5(1-0.166)}{0.166(1-0.5)} \end{aligned}$$

$$\text{SIM}(d6) = 0.301 + 0.301 + 0.698$$

$$\text{SIM}(d6) = 1.30$$

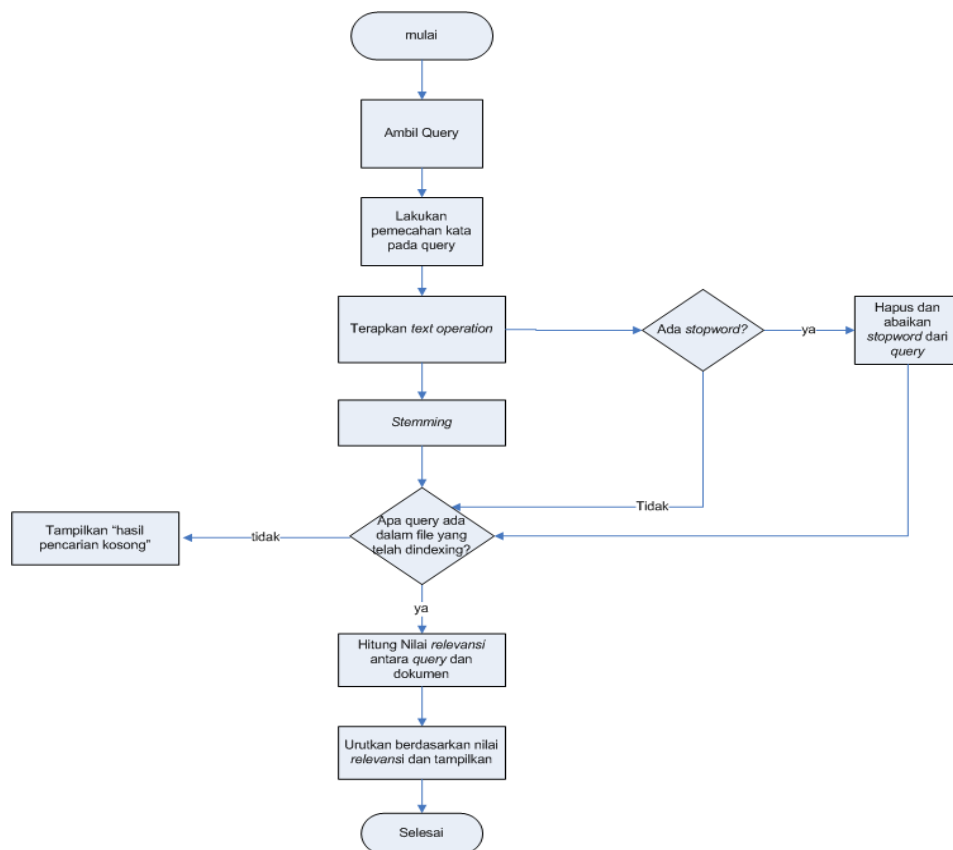
Berdasarkan perhitungan relevansi dokumen yang relevan, maka dokumen yang relevan dengan *query* adalah *dok 6, dok 3, dok 1, dok 2, dok 4, dok 5*.

4.2 Analisa sistem

setelah dilakukan analisa pada metode BIM, maka selanjutnya masuk pada tahapan analisa sistem, pembuatan *flowchart* dan juga *usecase diagram*.

4.2.1 Flowchart Sistem

Setelah dilakukan tahapan analisa sistem temu kembali informasi dengan *Binary Independence Model* yang akan dibangun, maka akan dilanjutkan dengan tahapan perancangan sistem. Sistem yang akan dirancang harus sesuai dengan hasil analisa sistem yang telah dilakukan pada subbab 4.1



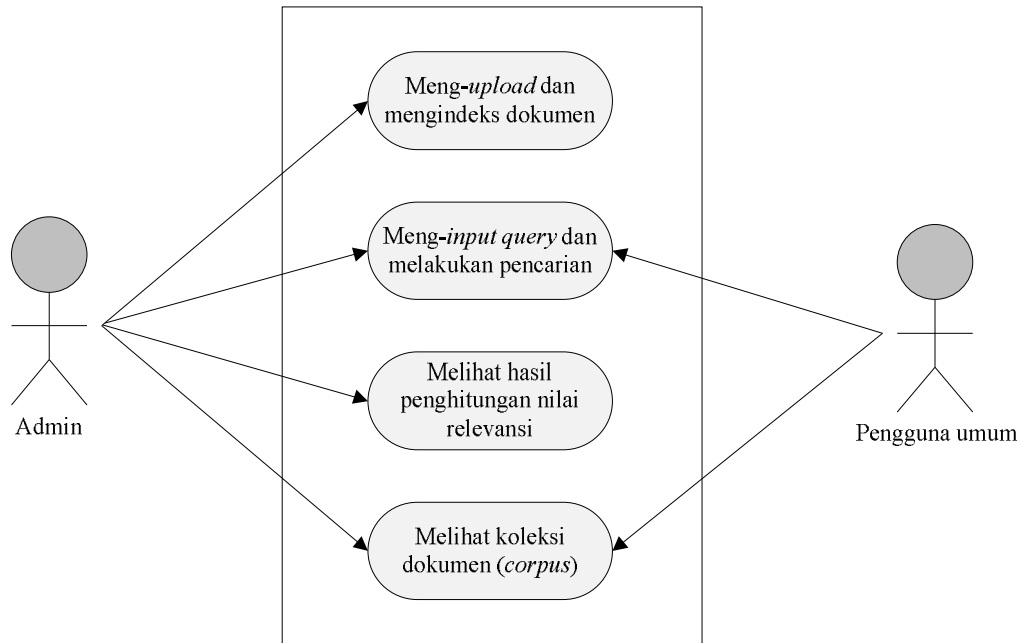
Gambar 4.1 *Flowchart* Sistem Temu Kembali Informasi, Perhitungan Nilai Relevansi Dan Perangkingan Dokumen

Pada perancangan sistem *information retrieval* terdapat beberapa tahapan-tahapan yang harus diikuti seperti gambar pada *flowchart* diatas. Di mulai dari *query* yang diinputkan oleh pengguna, setelah itu *query* yang diinputkan sistem akan melakukan pemecahan kata terhadap *query*, pemecahan ini dilakukan untuk

pemberian index terhadap *query*. Setelah dilakukan pemecahan kata, sistem akan melakukan pengecekan apakah terdapat *stopword* atau tidak, jika ada *stopword* sistem akan langsung menghilangkan kata-kata yang mengandung *stopword*, *stopword* ini telah disediakan sebelumnya dalam file *stopword.txt*. setelah proses *stopword* selesai dilakukan, dilanjutkan dengan pengecekan *Stemming* pada *query*, pengecekan dilakukan dengan mencari kata berimbuhan pada *database*, jika kata dasar tidak terdapat pada *database*, maka kata tersebut harus *distemming* terlebih dahulu, setelah dilakukan *stemming* maka dilanjutkan pengecekan *query* kepada file dokumen yang telah diindeks, pada penelitian ini dokumen yang telah diindeks dimasukkan pada file *token.txt*. Jika salah satu kata terdapat pada file *token.txt*, maka dilanjutkan dengan menghitung *relevansi query* terhadap dokumen yang tersedia, nilai *relevansi* yang dihasilkan akan dihitung menggunakan Metode *Binary Independence Model*. Setelah nilai *relevansi* didapatkan, maka sistem akan melakukan perangkingan dokumen yang dimulai dari nilai *relevansi* yang paling tinggi hingga nilai *relevansi* yang rendah.

4.2.2. Usecase Diagram

Use case diagram menjelaskan siapa-siapa saja *actor* yang terlibat atau dapat menggunakan sistem IR dengan *Binary Independence Model* ini dan apa saja yang dapat dilakukan (*scenario*) oleh *actor* terhadap sistem. *Actor* adalah semua yang berhubungan langsung ke sistem, yang memberi input atau menerima informasi dari sistem. *Use case diagram* untuk Sistem Temu Kembali Informasi dengan model *Binary Independence Model* yang akan dibangun dapat dilihat pada Gambar 4.4



Gambar 4.2. Use case diagram sistem IR dengan *Binary Independence Model*

Terdapat dua *actor* yang terlibat dalam sistem ini, yaitu *admin* dan pengguna umum. Ketika *login* ke sistem, *admin* akan diminta memasukkan *user id* dan *password*. Admin dapat melakukan empat skenario yaitu, meng-*upload* dokumen dan mengindeks dokumen di dalam *corpus*, menginputkan *query* dan menjalankan proses pencarian, admin juga dapat menampilkan nilai *relevansi* yang diperoleh dari hasil perhitungan sistem dengan *Binary Independence Model*, dan dapat melihat seluruh dokumen yang ada di dalam *corpus*. Sedangkan pengguna umum hanya dapat melakukan dua skenario yaitu, menginputkan *query* dan menjalankan proses pencarian, dan melihat seluruh dokumen yang ada di dalam *corpus*.

4.3 Perancangan

Pada tahapan ini, ada beberapa file yang perlu dirancang untuk tempat penyimpanan file seperti hasil *tokenization*, *stopword*, dan file-file lain yang mendukung untuk kinerja sistem, perancangan ini dinamakan perancangan *flatfile*. Selain itu, analisa yang lain yang dirasa perlu adalah analisa antar muka sistem.

4.3.1 Perancangan File teks (*flat file*)

Dalam perancangan sistem tugas akhir ini, untuk proses penyimpanan data maupun informasi tidak menggunakan *database relasional*, melainkan *flat file* yang menggunakan *file* teks (*plain text*) sebagai media penyimpanannya. Sehingga seluruh informasi dari hasil proses *tokenisasi*, pengindeksan (*indexing*), pembobotan lokal (*tf*), dan perhitungan nilai relevansi akan disimpan dalam *file* teks.

Daftar kata *stop-words* disimpan dalam *file Stopwords.txt* yang sebelumnya telah disimpan dalam pembangun sistem dan berjumlah 352 kata. Daftar *stop-words* ini berdasarkan sumber dari <http://www.scribd.com/doc/61824071/DAFTAR-PUSTAKA>. Setelah sistem dijalankan, informasi dari hasil proses *tokenisasi*, pengindeksan, pembobotan lokal (*tf*), dan perhitungan nilai relevansi akan disimpan ke dalam *file* *daftar_file.txt*, *token.txt*, dan *Nilai_relevansi.txt*.

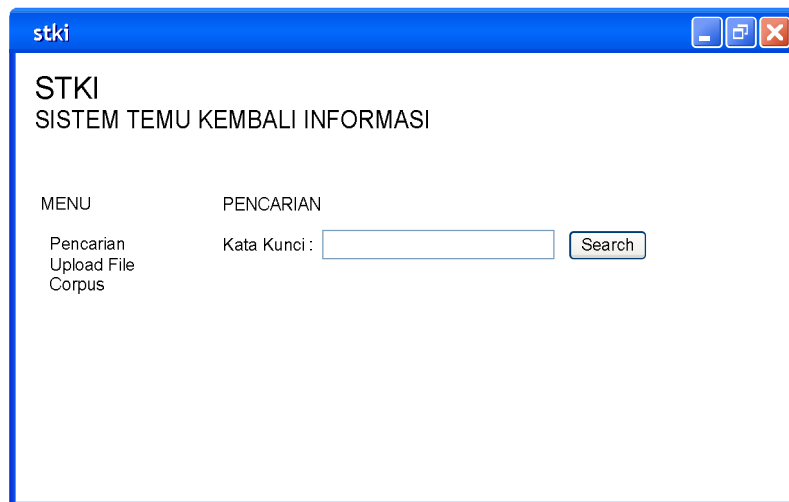
File *daftar_file.txt* akan menyimpan informasi berupa id dokumen, lokasi dokumen di dalam PC, alamat sumber asli dokumen, panjang dokumen, dan judul dokumen. Sedangkan *file* *token.txt* menyimpan informasi dari hasil *tokenisasi*, pengindeksan, dan pembobotan lokal, yaitu berupa kata (*term*) dan id dokumen terkait dimana kata tersebut muncul beserta *frekuensi* kemunculannya (*tf*) seperti yang terlihat dalam Tabel 4.2. *daftar_file .txt* akan menyimpan informasi berupa id dokumen dan nilai *relevansi* (korelevanan) dokumen terhadap *query* yang diinputkan pengguna menurut perhitungan *Binary Independence Model* yang telah diurutkan secara menurun (*descending*).

4.3.2 Perancangan Antarmuka (*Interface*) Sistem

Tahapan ini dilakukan dengan tujuan untuk dapat merancang antarmuka sistem yang akan dibangun dengan sebaik-baiknya sehingga sistem dapat menjadi *user friendly* bagi para penggunanya. *Interface* yang akan dibangun terdiri dari *form* tampilan utama (*form* pencarian), *form* login, *form* upload file, *form* corpus, *form* hasil pencarian, dan *form* proyeksi perhitungan.

1. *Form* tampilan utama (*form* pencarian)

Form tampilan utama sekaligus mencakup *form* pencarian, dimana *form* ini merupakan *form* utama ketika pengguna mulai menjalankan sistem. Rancangan *form* tampilan utama (*form* pencarian) dapat dilihat pada Gambar 4.5.

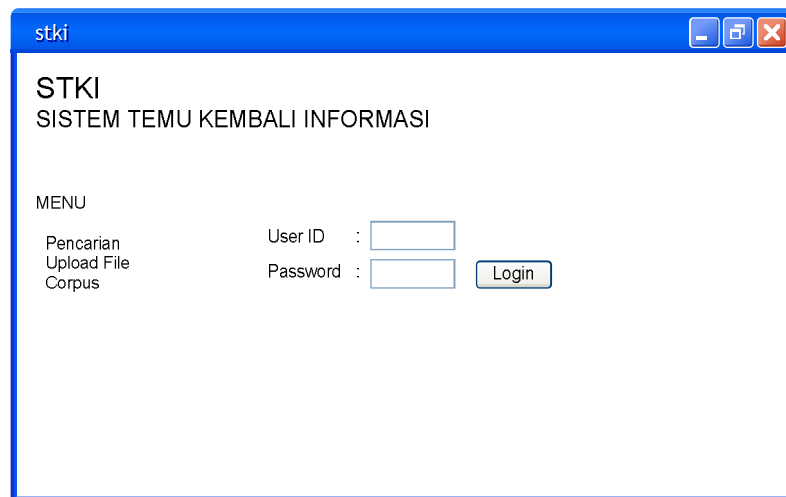
The image shows a screenshot of a software window titled 'stki'. Inside the window, the text 'STKI' and 'SISTEM TEMU KEMBALI INFORMASI' is displayed at the top. Below this, there are two main sections: 'MENU' and 'PENCARIAN'. The 'MENU' section lists three options: 'Pencarian', 'Upload File', and 'Corpus'. The 'PENCARIAN' section contains a text input field labeled 'Kata Kunci :', a 'Search' button, and a 'Search' button.

Gambar 4.3. Rancangan *form* tampilan utama (*form* pencarian)

2. *Form* login

Form login berfungsi ketika pengguna akan meng-*upload* dokumen yang secara otomatis juga akan menjalankan proses indeks dokumen (membangun *inverted index*), serta melihat hasil penghitungan nilai *relevansi* antara dokumen terhadap *query* yang diinputkan yang dilakukan oleh sistem. *Form*

login akan muncul ketika pengguna memilih menu *upload file*. Pengguna harus menginputkan *user id* dan *password*, ini artinya pengguna harus *login* sebagai admin. Adapun rancangan *form login* dapat dilihat pada Gambar 4.6.



The image shows a screenshot of a web application window titled "stki". The main heading is "STKI SISTEM TEMU KEMBALI INFORMASI". Below this, there is a "MENU" section with three items: "Pencarian", "Upload File", and "Corpus". To the right of the menu, there are two input fields labeled "User ID" and "Password", each followed by a colon. A "Login" button is positioned to the right of the "Password" field. The window has a blue border and standard Windows-style window controls (minimize, maximize, close) in the top right corner.

Gambar 4.4. Rancangan *form login*

3. *Form upload file*

Form upload file hanya bisa berfungsi ketika pengguna telah *login* sebagai admin, yaitu dengan menginputkan *user id* dan *password* pada *form login*. *Form upload file* berguna untuk proses meng-*upload* dokumen yang secara otomatis juga akan menjalankan proses *indexing* dokumen. Rancangan *form upload file* ditunjukkan pada Gambar 4.

stki

STKI
SISTEM TEMU KEMBALI INFORMASI

MENU

Pencarian
Upload File
Corpus
Logout

Upload File Baru :

Gambar 4.5. Rancangan *form upload file*

4. *Form corpus*

Ketika admin maupun pengguna umum ingin melihat keseluruhan dokumen yang ada di dalam koleksi dokumen, maka dapat memilih menu *corpus*. Dalam menu ini terdapat *form corpus* yang akan menampilkan isi koleksi dokumen keseluruhannya. Bentuk rancangan tampilan *form corpus* ditunjukkan pada Gambar 4.8.

stki

STKI
SISTEM TEMU KEMBALI INFORMASI

MENU

Pencarian
Upload File
Corpus

PENCARIAN

Kata Kunci :

Corpus (Koleksi Dokumen)

no_file	judul_file	cuplikan_file

Gambar 4.6. Rancangan *form corpus*

5. *Form* hasil pencarian

Form ini sebagai keluaran akhir (*output*) dari hasil pencarian oleh sistem. Pada bagian ini ditampilkan sejumlah dokumen yang relevan dengan *query* pengguna secara terurut *descending* sesuai hasil perhitungan sistem dengan menggunakan *Binary Independence model*. Rancangan tampilan untuk *form* hasil pencarian dapat dilihat pada Gambar 4.9.

stki

STKI
SISTEM TEMU KEMBALI INFORMASI

MENU PENCARIAN

Pencarian Kata Kunci :

Upload File

Corpus

Hasil Pencarian

no_file	judul_file	cuplikan_file
no_file	judul_file	cuplikan_file

[Tampilkan Proyeksi Perhitungan] (Jika sebagai admin)

Gambar 4.7. Rancangan *form* hasil pencarian

BAB V

IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi

Setelah dilakukan analisa dan perancangan terhadap sistem temu kembali informasi yang akan dibangun, maka tahap selanjutnya adalah tahap implementasi, yaitu tahap melakukan pengkodean (*coding*) terhadap hasil analisa dan rancangan sehingga akan diketahui apakah sistem telah menghasilkan tujuan yang diinginkan.

5.1.1 Batasan Implementasi

Batasan implementasi pada tugas akhir ini, yaitu koleksi dokumen (*corpus*) yang digunakan adalah dokumen yang berekstensi (*extension*) .htm, .html, .txt, dan .pdf.

5.1.2 Lingkungan Implementasi

Implementasi sistem dilakukan pada perangkat keras dan perangkat lunak dengan spesifikasi sebagai berikut:

1. Perangkat keras

Processor : Intel Pentium core i3, 2.1 GHz

Memory (RAM) : 2 GB

Harddisk : 320 GB

2. Perangkat lunak

Operating system : Windows 7 Ultimate

Bahasa pemrograman : PHP (*PHP Hypertext Preprocessor*)

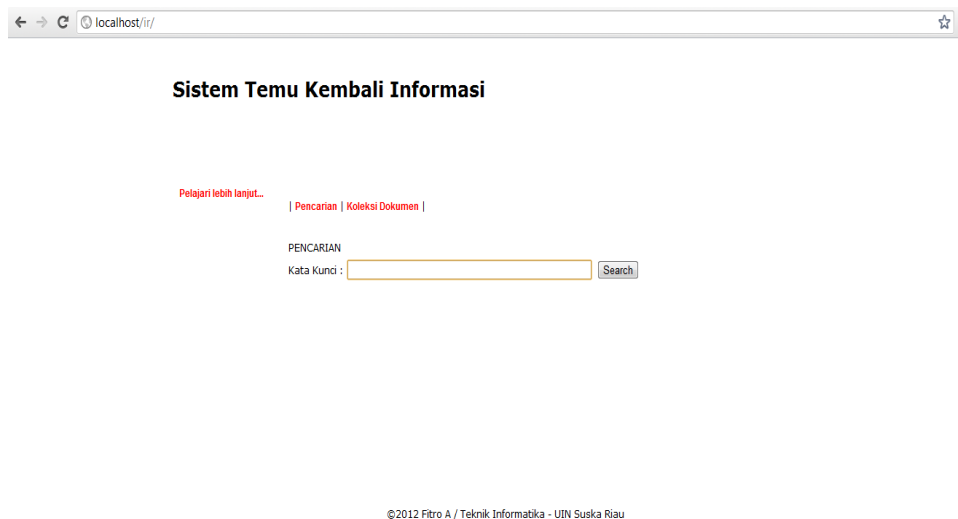
Web browser : Firefox

5.1.3 Hasil Implementasi

Dalam subbab ini ditampilkan hasil implementasi dari tahap pengkodean (*coding*) sesuai dengan analisa dan perancangan yang telah dilakukan Hasil implementasi meliputi, tampilan antarmuka sistem dan tampilan *file* yang menyimpan daftar *stop-words* dan informasi dari hasil proses *tokenisasi*, *indexing*, pembobotan *Biner*, dan perhitungan nilai *relevansi* (*rsv*).

1. Tampilan utama (menu pencarian)

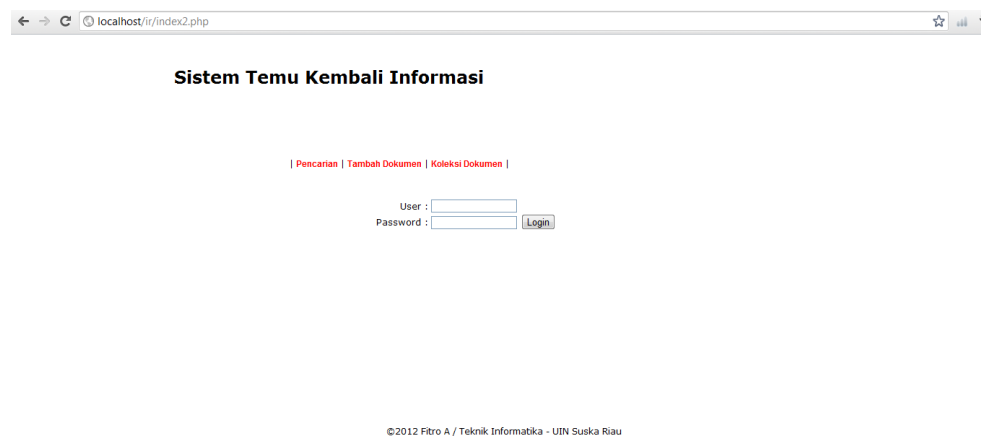
Tampilan utama yang juga sekaligus mencakup menu pencarian. Implementasi tampilan utama (menu pencarian) terlihat pada Gambar 5.1.



Gambar 5.1 Tampilan menu utama pencarian

2. Tampilan menu *login*

Menu *login* muncul ketika pengguna memilih menu *upload file*. Pengguna yang telah *login* sebagai admin dapat meng-*upload* dokumen yang secara otomatis juga menjalankan proses *indexing* (pembuatan *inverted index*). Tampilan menu *login* dapat dilihat pada Gambar 5.2.



Gambar 5.2 Tampilan menu login

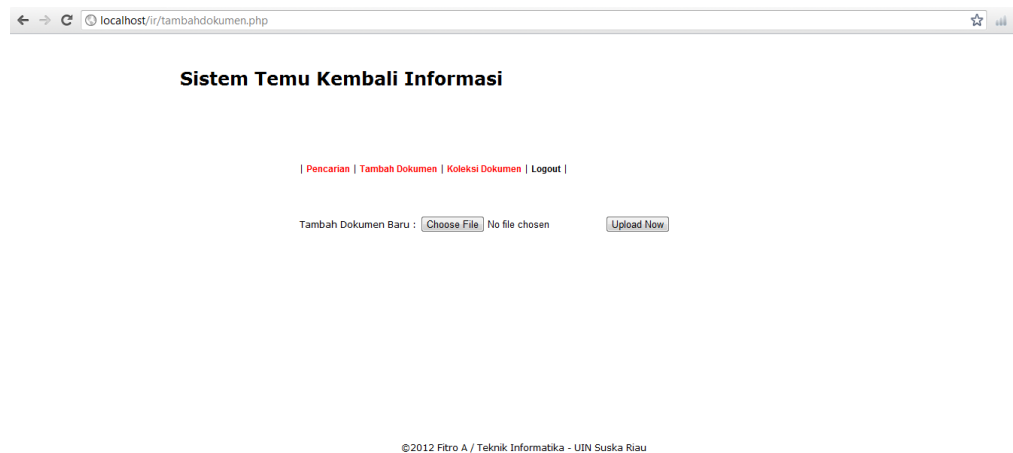
3. Tampilan menu *upload file*

Menu *upload file* hanya dapat berfungsi ketika pengguna telah *login* sebagai admin. Menu *upload file* berguna untuk proses meng-*upload* dokumen yang secara otomatis juga menjalankan proses indeks dokumen. Ketika proses *indexing* dijalankan, maka daftar *stop-words* yang dibutuhkan dalam pemrosesan *index* akan diambil ke dalam *file Stopwords.txt* yang telah disimpan sebelumnya dalam pembangun sistem. Setelah pengindeksan selesai, informasi dari hasil tokenisasi, pengindeksan, dan pembobotan biner disimpan ke dalam *Daftar_file.txt* dan *token.txt*. Tampilan menu *upload file*,

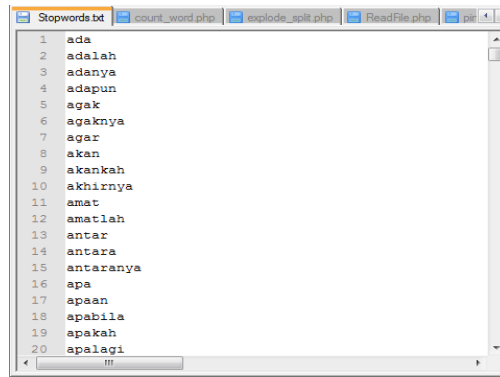
Stopwords.txt, Daftar_file.txt, dan Token.txt ditunjukkan pada Gambar 5.3, Gambar 5.4, Gambar 5.5, dan Gambar 5.6.

Dalam *file* Filelist.txt, setiap informasi yang disimpan dipisahkan oleh *delimiter* “|” (garis lurus) sebagai pembatas untuk dapat membedakan bagian-bagian informasi yang tersimpan tersebut. Sehingga format penyimpanannya pun menjadi:

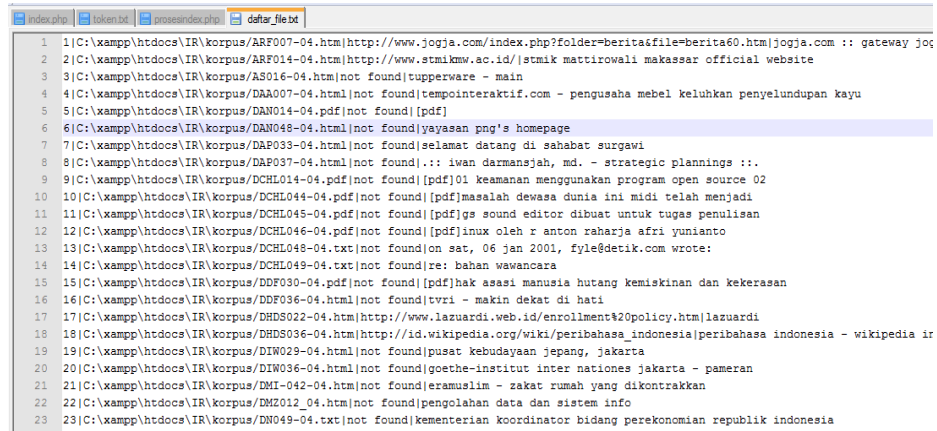
id dok | lokasi dokumen di dalam PC | link sumber asli dokumen | panjang dokumen | judul dokumen.



Gambar 5.3 Menu *upload* file



Gambar 5.4 Daftar stopwords.txt



Gambar 5.5 Daftar_file.txt

Untuk *file* Indexing.txt, informasi yang disimpan berupa kata dan id dokumen yang menunjukkan di dokumen mana saja kata tersebut muncul serta frekuensi kemunculannya. Informasi yang disimpan dipisahkan oleh *delimiter* “|” sebagai pembatas antara kata dan id dokumen. Antara id dokumen dan *tf* diberikan *delimiter* “:” (titik dua). Sedangkan untuk pembatas antara dokumen satu dan lainnya diberikan *delimiter* “,” (koma). Sehingga format penyimpanannya pun menjadi:

kata | id dok : frekuensi, id dok : frekuensi, id dok : frekuensi, dan seterusnya.

```

index.php | token.txt | prosesindex.php | daftar_file.txt | nilai_relevansi.txt
3278 diletakkan|10:1,12:1
3279 lokal|10:1,15:1,20:1,26:1
3280 extended|10:1,12:1
3281 key|10:1
3282 nilainya|10:1,21:1
3283 diterjemahkan|10:1
3284 berhenti|10:1,20:1,26:1
3285 escape|10:1
3286 mengakhiri|10:1
3287 looping|10:1
3288 pelepasan|10:1
3289 penutupan|10:1
3290 ambilparamchan|10:1,11:1
3291 chan|10:1,11:1
3292 ambilparameter|10:1,11:1
3293 md5|10:1
3294 chann|10:1
3295 function|10:1,11:1
3296 boolean|10:1,11:1
3297 uecx|10:1
3298 dec|10:1,11:1
3299 cnyxmb|10:1,11:1
3300 kirimsyrequest|10:1
3301 aktifkanintr|10:1,11:1
3302 bacabuffer|10:1,11:1

```

Gambar 5.6 Token.txt

4. Tampilan menu koleksi dokumen (*corpus*)

Admin maupun pengguna umum dapat melihat keseluruhan dokumen yang ada di dalam koleksi dokumen melalui menu *corpus*. Menu ini menampilkan judul dan cuplikan (*snippet*) dari masing-masing dokumen. Bentuk tampilan menu *corpus* seperti ditunjukkan pada Gambar 5.7.

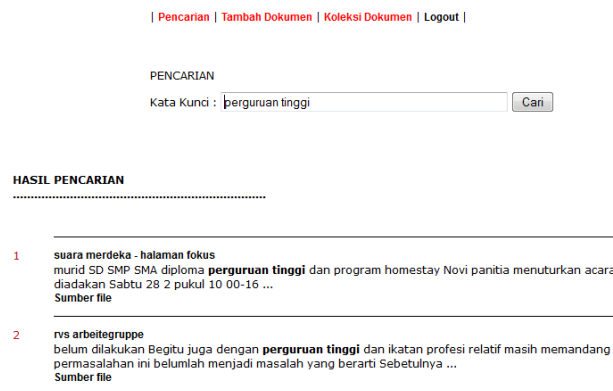


Gambar 5.7 Daftar koleksi dokumen

5. Tampilan hasil pencarian

Ketika pengguna menginputkan *query* dan memulai pencarian, maka sistem akan memulai pencarian dokumen berdasarkan tingkat relevansi dengan menggunakan metode binary independence model dan setelah itu menampilkan dokumen yang cari secara terurut menurun (*Descending*) dari nilai relevansi yang diperoleh. Menu ini menampilkan judul dan cuplikan (*snippet*) dari masing-masing dokumen. Hasil perhitungan nilai relevansi yang diperoleh sistem akan disimpan secara otomatis oleh sistem ke dalam *file* nilai_relevansi.txt. Bentuk tampilan hasil pencarian dan nilai_relevansi.txt dapat dilihat pada Gambar 5.8 dan Gambar 5.9.

Sistem Temu Kembali Informasi Menggunakan Metode Binary Independence Model



The screenshot shows a web interface for a search system. At the top, there are navigation links: "Pencarian", "Tambah Dokumen", "Koleksi Dokumen", and "Logout". Below these is a search bar labeled "PENCARIAN" with the text "Kata Kunci : perguruan tinggi" and a "Cari" button. The search results are titled "HASIL PENCARIAN" and are listed in a numbered format. The first result is from "suara merdeka - halaman fokus" and the second is from "rva arbeitgruppe".

| [Pencarian](#) | [Tambah Dokumen](#) | [Koleksi Dokumen](#) | [Logout](#) |

PENCARIAN

Kata Kunci : perguruan tinggi

HASIL PENCARIAN

1 suara merdeka - halaman fokus
murid SD SMP SMA diploma **perguruan tinggi** dan program homestay Novi panitia menuturkan acara diadakan Sabtu 28 2 pukul 10 00-16 ...
Sumber file

2 rva arbeitgruppe
belum dilakukan Begitu juga dengan **perguruan tinggi** dan ikatan profesi relatif masih memandang permasalahan ini belum menjadi masalah yang berarti Sebetulnya ...
Sumber file

Gambar 5.8 Hasil pencarian

Dalam *file* nilai_relevansi.txt, informasi yang disimpan dipisahkan oleh *delimiter* "|" sebagai pembatas antara id dokumen dan nilai relevansinya terhadap *query* yang diinputkan. Sehingga format penyimpanannya menjadi:

id dok | nilai relevansi

	index.php	token.bt	prosesindex.php	nilai_relevansi.bt	korpus.php	emptypage.php	kar
1	47#0.594747095742						
2	42#0.594747095742						
3	39#0.594747095742						
4	38#0.594747095742						
5	49#0.594747095742						
6	86#0.594747095742						
7	2#0.594747095742						
8	101#0.594747095742						
9	100#0.594747095742						
10	94#0.594747095742						
11	36#0.594747095742						
12	50#0.594747095742						
13	7#0.594747095742						
14	6#0.594747095742						
15	33#0.594747095742						
16	5#0.594747095742						
17	21#0.594747095742						
18	15#0.594747095742						
19	31#0.594747095742						
20	22#0.594747095742						
21	25#0.594747095742						
22	27#0.56066730617						
23	58#0.0340797895723						
24	57#0.0340797895723						
25	56#0.0340797895723						
26	52#0.0340797895723						
27	63#0.0340797895723						
28	51#0.0340797895723						
29	97#0.0340797895723						
30	99#0.0340797895723						

Gambar 5.9 Tampilan Nilai Relevansi

5.2 Pengujian Sistem

Dalam tahapan ini, sistem akan dijalankan dan diuji cobakan untuk mengetahui apakah sistem berjalan sesuai dengan hasil analisa dan tujuan yang diharapkan. Untuk mengetahui kemampuan sistem IR yang telah dibangun dalam tugas akhir ini, maka akan dilakukan pengujian dengan mengukur kualitas *retrieval*, yaitu dengan menghitung nilai *precision* dan *recall* dengan menggunakan *Stemming* dan tanpa menggunakan *Stemming*.

Query yang digunakan adalah:

Table 5.1 *Query* yang digunakan dalam pengujian

kesehatan masyarakat	<i>Query 1</i>
Ekonomi Indonesia	<i>Query 2</i>
Perkembangan Teknologi di indonesia	<i>Query 3</i>

5.2.1 Hasil Pengujian Menggunakan *Stemming*

Untuk memperoleh nilai *precision* dan *recall*, digunakan Persamaan 2.7, Persamaan 2.8. hasil perhitungan yang dihasilkan oleh metode binary independence model berdasarkan *query* yang diinputkan menggunakan *stemming* adalah:

1. *Query* : Kesehatan masyarakat indonesia

Berdasarkan *Query* yang diinputkan Ditunjukkan bahwa jumlah dokumen yang dikembalikan yang relevan dengan *query* (*tp*) sebanyak 90 dokumen, sedangkan dokumen yang tidak relevan (*fp*) sebanyak 87 dokumen. Dan untuk jumlah dokumen yang tidak dikembalikan yang relevan dengan *query* (*fn*) sebanyak 0 dokumen, sedangkan dokumen yang tidak relevan sebanyak (*tn*) 104 dokumen. Maka, nilai *precision* dan *recall* untuk *query* 1 adalah:

$$Precision \rightarrow P = tp / (tp + fp) = 90 / (90+87) = 90/177 = 0,50$$

$$Recall \rightarrow R = tp / (tp + fn) = 90 / (90+0) = 90/90 = 1$$

Table 5.2 *Query* “kesehatan masyarakat”

Keterangan	<i>Relevant</i>	<i>Nonrelevant</i>
<i>Retrieved</i>	90	87
<i>Not retrieved</i>	0	104

2. *Query* : Ekonomi Indonesia

Berdasarkan *Query* yang diinputkan ditunjukkan bahwa jumlah dokumen yang dikembalikan yang relevan dengan *query* (*tp*) sebanyak 112 dokumen, sedangkan dokumen yang tidak relevan (*fp*) sebanyak 101 dokumen. Dan untuk jumlah dokumen yang tidak dikembalikan yang relevan dengan *query* (*fn*) sebanyak 0 dokumen, sedangkan dokumen yang tidak relevan sebanyak (*tn*) 68 dokumen. Maka, nilai *precision* dan *recall* untuk *query* 2 adalah:

$$Precision \rightarrow P = tp / (tp + fp) = 112 / (112+101) = 112/213 = 0,52$$

$$Recall \rightarrow R = tp / (tp + fn) = 112 / (112+0) = 90/90 = 1$$

Table 5.3 Query “Ekonomi Indonesia”

Keterangan	Relevant	Nonrelevant
<i>Retrieved</i>	112	101
<i>Not retrieved</i>	0	68

3. Query : Perkembangan Teknologi di indonesia

Berdasarkan Query yang diinputkan, ditunjukkan bahwa jumlah dokumen yang dikembalikan yang relevan dengan query (*tp*) sebanyak 115 dokumen, sedangkan dokumen yang tidak relevan (*fp*) sebanyak 109 dokumen. Dan untuk jumlah dokumen yang tidak dikembalikan yang relevan dengan query (*fn*) sebanyak 0 dokumen, sedangkan dokumen yang tidak relevan sebanyak (*tn*) 57 dokumen. Maka, nilai *precision* dan *recall* untuk query 2 adalah:

$$Precision \rightarrow P = tp / (tp + fp) = 115 / (115+109) = 115/224 = 0,51$$

$$Recall \rightarrow R = tp / (tp + fn) = 115 / (115+0) = 115/115 = 1$$

Table 5.4 query “Perkembangan Teknologi di indonesia ”

Keterangan	Relevant	Nonrelevant
<i>Retrieved</i>	115	109
<i>Not retrieved</i>	0	57

5.2.2 Hasil Pengujian Tanpa Menggunakan Stemming

Berikut adalah hasil pengujian tanpa menggunakan stemming:

1. Query : Kesehatan masyarakat indonesia

Berdasarkan Query yang diinputkan, ditunjukkan bahwa jumlah dokumen yang dikembalikan yang relevan dengan query (*tp*) sebanyak 83 dokumen, sedangkan dokumen yang tidak relevan (*fp*) sebanyak 89 dokumen. Dan untuk jumlah dokumen yang tidak dikembalikan yang relevan dengan query (*fn*) sebanyak 7 dokumen, sedangkan dokumen yang tidak relevan sebanyak (*tn*) 102 dokumen. Maka, nilai *precision* dan *recall* untuk query 1 adalah:

$$Precision \rightarrow P = tp / (tp + fp) = 83 / (83+89) = 83/172 = 0,4$$

$$Recall \rightarrow R = tp / (tp + fn) = 83 / (83+7) = 83/90 = 0.92$$

Table 5.5 *Query* “kesehatan masyarakat”

Keterangan	<i>Relevant</i>	<i>Nonrelevant</i>
<i>Retrieved</i>	83	89
<i>Not retrieved</i>	7	102

2. *Query* : Ekonomi Indonesia

Berdasarkan *Query* yang diinputkan ditunjukkan bahwa jumlah dokumen yang dikembalikan yang relevan dengan *query* (*tp*) sebanyak 112 dokumen, sedangkan dokumen yang tidak relevan (*fp*) sebanyak 101 dokumen. Dan untuk jumlah dokumen yang tidak dikembalikan yang relevan dengan *query* (*fn*) sebanyak 0 dokumen, sedangkan dokumen yang tidak relevan sebanyak (*tn*) 68 dokumen. Maka, nilai *precision* dan *recall* untuk *query* 2 adalah:

$$Precision \rightarrow P = tp / (tp + fp) = 112 / (112+101) = 112/213 = 0,52$$

$$Recall \rightarrow R = tp / (tp + fn) = 112 / (112+0) = 90/90 = 1$$

Table 5.6 *Query* “Ekonomi Indonesia”

Keterangan	<i>Relevant</i>	<i>Nonrelevant</i>
<i>Retrieved</i>	112	101
<i>Not retrieved</i>	0	68

3. *Query* : Perkembangan Teknologi di indonesia

Berdasarkan Tabel 5.6, ditunjukkan bahwa jumlah dokumen yang dikembalikan yang relevan dengan *query* (*tp*) sebanyak 56 dokumen, sedangkan dokumen yang tidak relevan (*fp*) sebanyak 72 dokumen. Dan untuk jumlah dokumen yang tidak dikembalikan yang relevan dengan *query* (*fn*) sebanyak 18 dokumen, sedangkan dokumen yang tidak relevan sebanyak (*tn*) 135 dokumen. Maka, nilai *precision* dan *recall* untuk *query* 3 adalah:

$$Precision \rightarrow P = tp / (tp + fp) = 56 / (56+72) = 56/128 = 0,43$$

$$Recall \rightarrow R = tp / (tp + fn) = 56 / (56+18) = 56/74 = 0.75$$

Table 5.7 *Query* “Perkembangan Teknologi di indonesia ”

Keterangan	<i>Relevant</i>	<i>Nonrelevant</i>
<i>Retrieved</i>	56	72
<i>Not retrieved</i>	18	135

5.2.3 Pengujian Tipe Dokumen

Pengujian tipe dokumen dilakukan untuk mengetahui apakah dokumen-dokumen dengan ekstensi .htm, .html, .txt, dan .pdf yang digunakan dalam penelitian sistem temu kembali informasi dengan *Binary Independence Model* ini berhasil atau tidak di dalam pemrosesan sistem yang dibangun. Tabel 5.6 menunjukkan hasil dari pengujian tipe dokumen yang dilakukan.

Tabel 5.8 Pengujian Tipe Dokumen

No	Prosedur Pengujian	Tipe Dokumen (<i>extension</i>)			
		.htm	.html	.txt	.pdf
1	Membuka dan membaca isi dokumen	Berhasil	Berhasil	Berhasil	Berhasil
2	Penghapusan <i>tag markup</i> dan format khusus dari dalam dokumen	Berhasil	Berhasil	-	-
3	Tokenisasi isi dokumen	Berhasil	Berhasil	Berhasil	Berhasil
4	Penghapusan <i>stop-words</i> dari dalam dokumen	Berhasil	Berhasil	Berhasil	Berhasil
5	Pengindeksan dokumen	Berhasil	Berhasil	Berhasil	Berhasil
6	Menampilkan isi dokumen	Berhasil	Berhasil	Berhasil	Berhasil

5.2.4 Kesimpulan Pengujian

Berdasarkan pengujian yang telah dilakukan, maka dapat diambil kesimpulan:

1. Untuk pengujian *Binary Independence Model* menggunakan stemming, persentase kualitas *retrieval* untuk *query* 1 terhadap jumlah dokumen yang berhasil di-*retrieve* oleh sistem, yaitu *precision* 0,51 dan *recall* 1. Sedangkan persentase kualitas *retrieval* untuk *query* 2 berdasarkan jumlah dokumen yang berhasil di-*retrieve*, yaitu *precision* 0.52 dan *recall* 1. Dan persentase kualitas

retrieval untuk *query* 3 berdasarkan jumlah dokumen yang di-*retrieve*, yaitu *precision* 0,51 dan *recall* 1.

2. Untuk pengujian *Binary Independence Model* Tanpa menggunakan *stemming*, persentase kualitas *retrieval* untuk *query* 1 terhadap jumlah dokumen yang berhasil di-*retrieve* oleh sistem, yaitu *precision* 0,4 dan *recall* 0,92. Sedangkan persentase kualitas *retrieval* untuk *query* 2 berdasarkan jumlah dokumen yang berhasil di-*retrieve*, yaitu *precision* 0,52 dan *recall* 1. Dan persentase kualitas *retrieval* untuk *query* 3 berdasarkan jumlah dokumen yang di-*retrieve*, yaitu *precision* 0,43 dan *recall* 0,75.
3. Sistem Temu Kembali Informasi dengan menggunakan metode *BIM* lebih banyak menemukan dokumen yang *relevan* dan dokumen yang di *retrieve* dibantu dengan menerapkan *algoritma stemming* nazief adriani dalam sistem pencarian.
4. Berdasarkan penelitian yang telah dilakukan, Kerelavan sebuah dokumen yang dicari, ditentukan oleh pengguna sendiri. Sistem hanya mengembalikan dokumen yang berhubungan dengan *query* yang diinputkan oleh pengguna.

BAB VI

PENUTUP

6.1 Kesimpulan

Berdasarkan tahapan-tahapan yang telah dilakukan dari tahapan pengumpulan data, analisis dan perancangan serta pengujian terhadap sistem temu kembali informasi, maka dapat diambil kesimpulan :

1. Penambahan *stemming* pada penerapan metode *Binary Independence Model* dapat meningkatkan kemampuan sistem dalam menemukan dokumen.
2. Penerapan model ini menghasilkan nilai relevansi yang hampir sama pada saat perangkingan.
3. Sistem Temu Kembali Informasi dengan penerapan Metode *Binary Independence model* dapat *meretrieve* dokumen yang *relevan* dan tidak *relevan*, sehingga dapat membantu pengguna dalam menemukan dokumen yang dibutuhkan.

6.2 Saran

Berdasarkan hasil pengujian yang dilakukan pada penelitian ini serta kesimpulan diatas, dapat disampaikan saran-saran untuk perbaikan pada pembangunan penelitian selanjutnya, yaitu :

1. Pada penelitian ini berdasarkan hasil pengujian, *Metode Binary Independence model* masih memiliki nilai relevansi yang sama, misalnya antara dokumen pada rangking 1 dan rangking 2 karena menggunakan pembobotan biner. Oleh karena itu, pada penelitian selanjutnya agar pembobotan yang dilakukan bisa menggunakan metode lain misalnya menggabungkan metode Ruang Vector dengan metode *Binary Independence Model*.

2. Penelitian ini menggunakan metode *Binary Independence Model*, pada penelitian selanjutnya agar dapat menggunakan metode lain yang terdapat *information retrieval*, seperti *Relevance Feedback*, *Naïve Bayes*, atau juga dapat dikembangkan dengan berbagai metode lain yang berhubungan dengan *information retrieval*.

DAFTAR PUSTAKA

- Agusta, L.. Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief dan Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia, 2009.
- Cios, Krzysztof J. Etc. Data Mining A Knowledge Discovery Approach, Springer. 2007
- Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze,. *Probabilistic information Retrieval*, page 222-225, 2009.
- Fuhr, N.. *Probabilistic Models in Information Retrieval*. The Computer Journal 35(3), pages 243-255,1992.
- Manning, Christoper D, Ragnavan Prabhakar, Schutze, Hinrich, *Introduction to Information Retrieval*, Cambridge University Press, 2009.
- Ramadhany, Taufik, *analisa dan implementasi penerapan algoritma genetika pada sistem temu kembali informasi*. Teknologi Bandung,2008
- Thomas Roelleke, Jun Wang, *Probabilistic Logical Modelling Of The Binary Independence Retrieval Model*,University of London,2007.